2000

Virtual production system

Kuo-Cheng Ko
Iowa State University

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

Virtual production system

by

Kuo-Cheng Ko

A dissertation submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of
DOCTOR OF PHILOSOPHY

Major: Industrial Engineering
Major Professor: Dr. Pius J. Egbelu

Iowa State University
Ames, Iowa
2000

UMI Number: 9962826

Bell & Howell Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

Graduate College

Iowa State University

This is to certify that the Doctoral dissertation of

Kuo-Cheng Ko

has met the dissertation requirements of Iowa State University

Signature was redacted for privacy.

Committee Member

Signature was redacted for privacy.

Committee Member

Signature was redacted for privacy.

Committee Member

Signature was redacted for privacy.

Committee Member

Signature was redacted for privacy.

Major Professor

Signature was redacted for privacy.

For the Major Program

Signature was redacted for privacy.

For the Graduate College

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to express my deep appreciation to my major professor, Dr. Pius J. Egbelu, who not only worked very closely with me at every stage of this research but also helped me to build up my confidence and engineering profession. His inductive and friendly advice made possible the completion of the dissertation.

I gratefully acknowledge the valuable suggestions and comments given to me by Dr. Doug Gemmill, Dr. Timothy Van Voorhis, Dr. John Wacker, and Dr. Yoshinori Suzuki. Special thanks are due Dr. Jo Min and Dr. Sarah Ryan for their unselfish assistance. I thank Lindo System Inc. for loaning me the commercial version of LINGO. My appreciation also goes to the other faculty members and my fellow graduate students of the Department for making my study at Iowa State University very pleasant. I thank the IMSE office staff for their administrative and facility assistance.

I would like to thank my parents, Ching-Chang Ko and Shin Kao, and my brother, Kuo-Ping Ko, for their support and encouragement throughout my study period and for giving me the opportunity to pursue my engineering education in United States. I appreciate my parents-in-law, Ging-Han Hwang and Fang-Hui Li, for their kindness and favors. My special gratitude goes to my wife, Yun-Yih Hwang. Her understanding and patient allow me to focus on my research. Her brightly smile and delicious food recharge me and make me go forward. This dissertation is dedicated to my parents, parents-in-law, and lovely wife.

ABSTRACT

To satisfy customer's demands in today's market, industry and academe have invested considerable effort to make production systems more efficient and competitive. The production systems that have been implemented in industry have their own unique advantages under certain conditions. In practice, once a production system is adopted in a shop, the operation mode of the shop will remain the same over time. However, in the face of facing a changing product mix environment, a shop needs an adaptable production system to gain the best performance possible.

The objective was to develop a systematic procedure to construct a virtual production system that allows a shop to switch from one operation mode to another without physical reconfiguration of the shop. The machines and the material handling system of a shop could be logically reorganized into various patterns to obtain different versions of a virtual production system, which actually exists as a set of information in a computer database. A reconfiguration of the data in the database leads to a corresponding logical reorganization of the physical system. Hence, on the one hand, the shop layout remains the same, while on the other hand, the mode of operation of the shop can be changed logically over time.

In this thesis, the performance of virtual production systems and other production systems are examined with an experiment involving three different measures. The results obtained show that virtual production systems are superior to traditional production systems and are competitive to those production systems with movable machines. However, when considering the cost incurred to reconfigure a shop physically and the fact that movable machines are not usually employed in most industries, the virtual production system provides a feasible and reasonable means to improve a shop's performance in a dynamic changing product mix environment.

CHAPTER 1. INTRODUCTION

The objective of this research is to develop a systematic procedure that allows an existing batch manufacturing shop to adapt its mode of operation in response to a changing product mix in order to optimize some measure of performance. The mode of operation refers to the type of production system employed by the shop to accomplish its production objectives. It is assumed that the shop already exists and has a job shop type layout in which the machines cannot be moved to obtain a new physical layout to adapt to a changing product mix. However, the machines and the material handling system in the shop can be reorganized logically into various patterns to obtain different versions of virtual production system. During each production session, depending on the product mix, a decision must be made between operating the system as a job shop or operating it as one of various versions of virtual production system. The aim of the technique developed in this research is to identify the form of production system, under a given production scenario, that would result in the best performance for the shop in terms of total machine setup and material handling time/cost.

A virtual production system, which is a logical rather than a physical arrangement or organization of the machines and material handling system in a manufacturing shop, exists as a set of information in a computer database. A reconfiguration of the data in the database leads to a corresponding logical reorganization of the physical system. A logical arrangement of a shop is equivalent to one virtual way to organize the shop. Thus, a logical arrangement corresponds directly to one view of a shop, which is how the shop is seen in a computer database with respect to how the production resources are organized or related to one another.

1.1 Production Systems

For many years, industry and academe have invested considerable effort to make production systems more efficient and adaptive to changes in demand and technology. If production systems are classified according to the arrangement of machines and departments in a plant, production systems in industry fall into four major categories: job shop, mass and

continuous production, batch production, and traditional cellular manufacturing [1]. In addition, virtual cellular manufacturing, which emerged in the 1980s, has become a subject of academic research. It is claimed that a virtual cellular manufacturing system might combine the advantages of job shop and traditional cellular manufacturing to produce a fifth category. The five production systems are described in the following sections.

1.1.1 Job Shop

The main characteristic of a job shop is that it produces a wide variety of products in relatively small volume [1, 2]. Machines with the same functions are arranged together to form a department in a plant, as shown in Figure 1. An item being produced will jump from one department to another based on the item's operational sequence. Because a job shop type factory is designed to produce a variety of different products, it must have relatively high flexibility. In general, a job shop is very adaptive to a dynamic environment in which the product types and desired volumes change frequently. An estimated 30 to 50 percent of the manufacturing systems in the United States are of the job shop type [1]. However, the frequency of machine set-ups and excessive material handling between departments in a job shop result in lower productivity. In addition, high expense may be associated with the large variety of tools and fixtures.

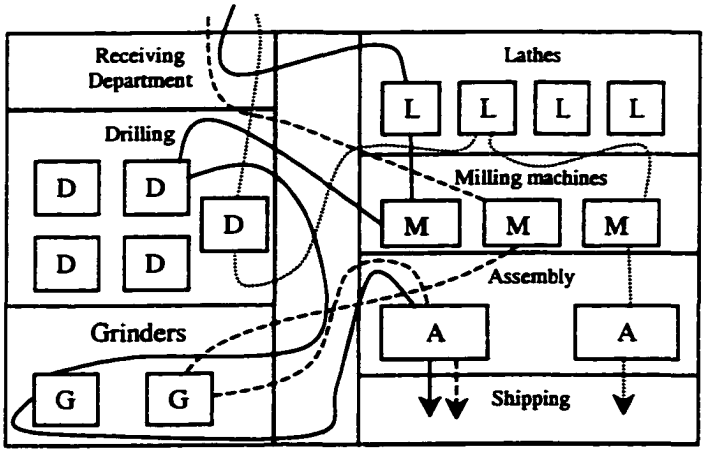


Figure 1. The layout for job shop

1.1.2 Mass Production System

A mass production system produces few products in large volumes [1, 2]. To produce a large volume of a product type, the machines needed for production are arranged sequentially and organized together to form a dedicated production line, as shown in Figure 2. In most cases, the machines in a production line need to be set up only once. The flow in a mass production system is much smoother than that of any other production system; as a result, the mass production system has the highest productivity. However, because a production line is employed for only one or very few product types, a mass production system is relatively inflexible. Because, a mass production system is the least adaptive production system of all, it is unsuitable for a production environment that experiences a changing product mix.

1.1.3 Batch Production System

The batch production system lies between the job shop and mass production systems [1, 2]. The main characteristic of a batch production system is that it produces a range of products, each one in medium volume. The layout of a batch production system is functionally similar to that of a job shop, as shown in Figure 3. Because products are processed in batches, some of the recurring fixed costs between individual batches can

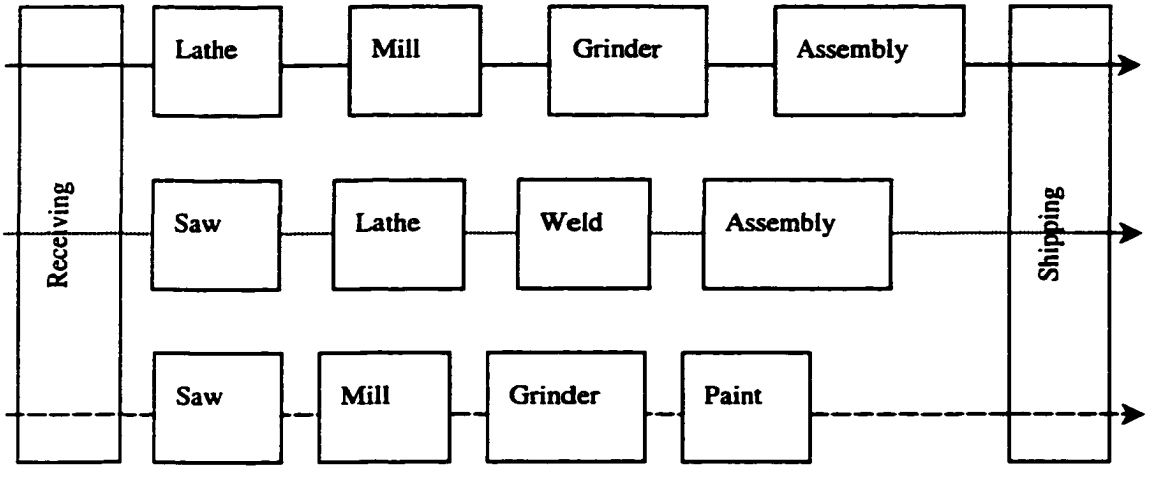


Figure 2. The layout for mass production

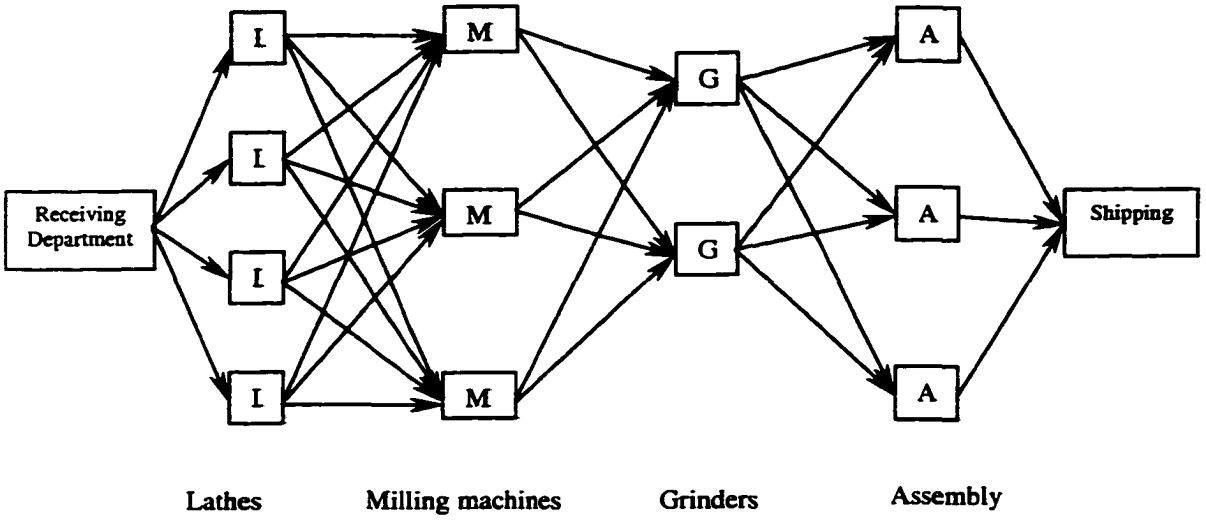


Figure 3. The layout for batch production

be shared. However, the handling of a large amount of work in progress and finished products in a shop are major concerns. Also, because products are produced in batches, a single product may occupy a machine for a considerable time, which might necessitate long delays in working on other products. Moreover, in a batch production system, the products and the relative sizes of batches to be produced during a period (usually a quarter) are generally known ahead of time, a factor that cannot be applied in a changing product mix environment. In general, the batch production system is most suitable for a company that produces and markets mature products with stable periodic demands.

1.1.4 Cellular Manufacturing System

The main characteristic of cellular manufacturing system is that it groups together the machines required to produce a family of parts [3]. The typical layout of a traditional cellular manufacturing system is shown in Figure 4. Jobs in the same part family could share the same setup, or a common setup could be designed for the whole part family. With the common setup, accomplishing each job in a part family needs only a minor setup, so that the set-up time for producing the whole part family is reduced significantly.

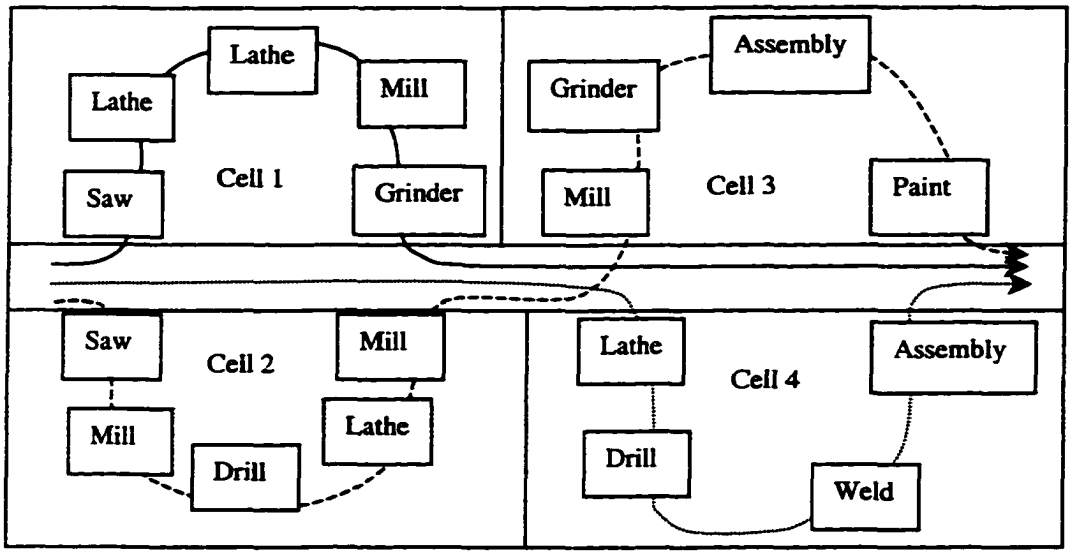


Figure 4. The layout for cellular manufacturing

Ideally, a part family is entirely completed within its own dedicated manufacturing cell. As a result, the productivity of a traditional cellular manufacturing system is higher than that of a job shop type factory. In addition, because manufacturing cells are formed and organized according to part families, cellular manufacturing systems are more flexible than mass production systems.

Although cellular manufacturing systems offer some benefits, several issues need to be addressed. First, in a traditional cellular manufacturing system, a machine cannot be shared by multiple cells; that is, a machine resides in one cell only. When a machine is required by more than one cell, a decision must be made on whether to duplicate the machine or not. At this point, many factors must be considered, such as space, cost, machine utilization, and so forth. In practice, it is not unusual for a product to visit more than one cell to be finished [4].

Second, in a traditional cellular manufacturing system design, the operational sequences of parts are usually ignored. As a result, part movement must be considered. Furthermore, in a changing product mix environment, implementation of cellular manufacturing will call for frequent machine reconfiguration. As a result, the machine relocation problem is encountered. Again, many factors need to be considered and resolved before the machines

are physically relocated. These factors include the weight and size of machines, the cost and time involved in moving them, and so forth. In view of these concerns, cellular manufacturing does not seem to be an appropriate production system in a dynamic environment, .which reduces both the flexibility and the machine utilization of a shop in traditional cellular manufacturing [4].

The contrast between job shop and cellular manufacturing has been the subject of many discussions. Based on some controlled experiments, it has been concluded that a cellular manufacturing system is superior to a job shop, or *vice versa,* as shown in Table 1. In different papers, job shop and cellular manufacturing have been reported to have the same advantages, such as average work-in-process level and average flow time, as summarized in Table 1.

In general, the job shop performs better than cellular manufacturing with regard to queue-related variables, such as average queue length and average waiting time [5, 6]. On the other hand, cellular manufacturing has the advantages of shorter setup time and shorter travel distance [5, 6, and 10-13]. In addition, some researchers have provided further insights into the parameter ranges or conditions in which either cellular manufacturing or the job shop could be superior [14-19].

1.1.5 Virtual Cellular Manufacturing System

A new concept, the virtual manufacturing cell, was first proposed by McLean *et al.* in 1982 [21]. The production system based on this concept, Virtual Cellular Manufacturing, is

Table 1. Job shop production *vs.* cellular manufacturing

	Job Shop Production	Cellular Manufacturing
Reported Superior Performance	Average queue length [5, 6]	Average setup time [5, 6, 10-13]
	Average waiting time [5, 6]	Average traveling distance [5, 6, 10, 11]
	Average work-in-process level [5-8]	Average work-in-process level [10, 11]
	Average flow time [5-9]	Average flow time [10-12]

very similar to a traditional cellular manufacturing system; machine cells and part families are also applied in virtual cellular manufacturing. However, virtual cellular manufacturing requires different cell configuration from that of traditional cellular manufacturing.

Virtual cells have three characteristics that distinguish them from traditional machine cells: a) they are logical, not physical, b) they are adaptable, and c) they allow machine and cell sharing. Unlike a traditional cell, which is a physical entity, a virtual cell is a logical entity. A virtual cell defines its grouping of machines in a computer; in other words, machines are not physically moved but are conceptually grouped together to form a virtual cell. Therefore, a virtual cell is no longer identifiable as a fixed physical machine cell. Machines belonging to the same virtual cell during any period may not necessarily occupy the same geographic region of a shop floor.

Because virtual cells are adaptable, they are very suitable for a dynamic changing product mix environment. Virtual cells are formed in response to the product mix released for production during a production session. Once a batch of jobs is completed and another batch is released for production, a new set of virtual cells may be reconfigured. Thus, machine set that constitutes a cell constantly changes as the product mix changes.

The machine-sharing concept is applied among virtual cells. In virtual cellular manufacturing, connections between machines are accomplished by a highly automated material handling system [4, 22]. As a result, not only is it unnecessary to change a shop's current layout, but a machine can serve more than one virtual cell.

An example of virtual cells is shown in Figure 5. Virtual cell 1 consists of four machines, L, M, D, and G. Virtual cell 2 and virtual cell 3 share the same machine, machine G. According to McLean *et al.* [21], virtual manufacturing cells provide more flexibility than traditional manufacturing cells by sharing machines; this suggests that virtual cellular manufacturing might have a bright future. However, a procedure for configuring virtual cells has rarely been reported in the literature to date.

The main drive in implementation of virtual cellular manufacturing is to take advantage of the benefits of traditional cellular manufacturing, while avoiding the disadvantage of locking the shop into fixed cells. Certain conditions favor the adoption of the concept of virtual cells; they are:

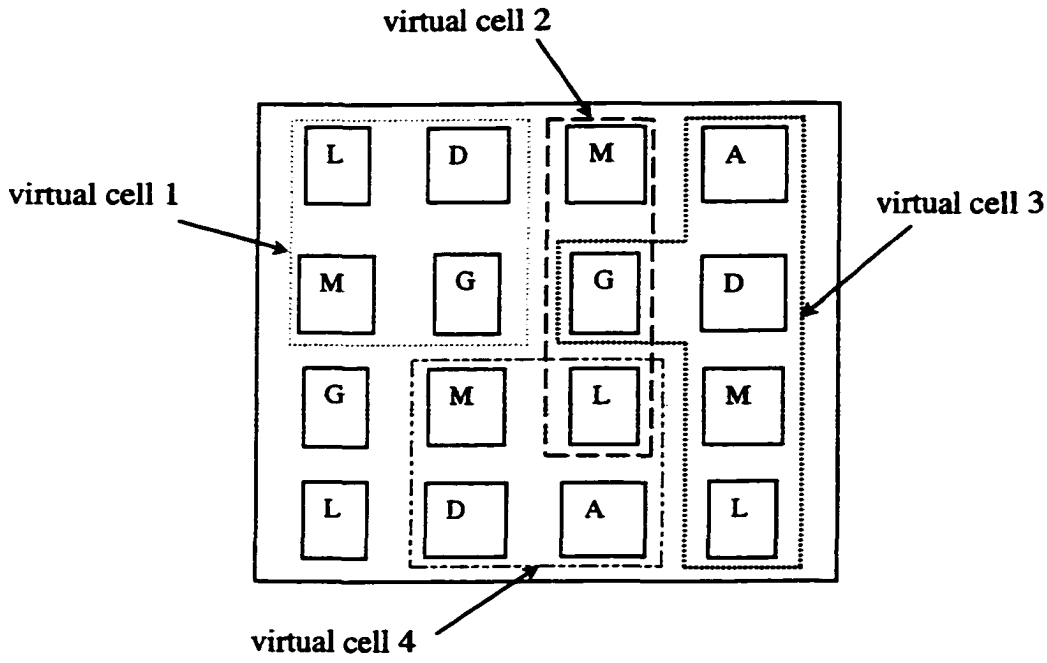


Figure 5. The layout of virtual cellular manufacturing

(1) production environments where the product mix is unstable or changing;
(2) shops in which transforming the existing layout to a cell layout may be very expensive or disrupt operations; and
(3) production environments with existing layouts in which the machines are too heavy and bulky to move or machines cannot be put together because of incompatibility.

As previously stated, the use of each kind of production system has its own pros and cons. The job shop and mass production systems represent two systems at opposite ends. The batch production system is more suitable for a stable production environment than for a dynamic production environment. Cellular manufacturing is a compromise between job shop and mass production, but presents numerous challenging issues. Virtual cellular manufacturing tries to offer the advantages of cellular manufacturing in a job shop type plant. Unfortunately, a virtual cell formation procedure is not yet available.

For a changing product mix environment, it seems unlikely that, given the changing product mix, one particular form of production system will consistently perform at a superior level. What is needed is a procedure that can identify the shop arrangement or production system type that would yield the best system performance. The development of such a procedure motivated this research. In this research, it is assumed that a job shop production system currently exists. However, the shop can also be reconfigured into one of different versions of virtual production system during any production instance. A choice of production system during a given instance is to be made from among the job shop and the various forms of virtual production systems. The various forms of virtual production system will be presented in succeeding sections.

1.2 Virtual Production System

The employment of a virtual production system in operating a manufacturing shop is proposed. It is assumed that the shop already exists and that the machines in the shop are arranged according to a job shop or process layout. Because of the size of the machines or other operating constraints, physical reconfiguration of the layout is not feasible, i.e., the physical layout is fixed. It is further assumed that a mobile material handling system such as the automated guided vehicles system (AGVS) is used for handling parts in the shop. In the AGVS, the direction of traffic flow on the guidepath segments can be reversed by manipulating data in a database through a computer software. The tasks to be performed in this research are to develop a procedure for generating the possible virtual production system configurations and to select the best configuration to use in a given production scenario involving an instance of product mix. The existing job shop configuration and the various virtual production system configurations constitute the set of production systems from which the best system is to be selected.

A virtual production system is composed of two modules, the processing system configuration module and the networking module, as shown in Figure 6. The processing system configuration module is concerned with how machines in the shop are organized.

In contrast, the networking module is focused on how the material handling system is organized. In a dynamic production environment, both modules may need to be updated

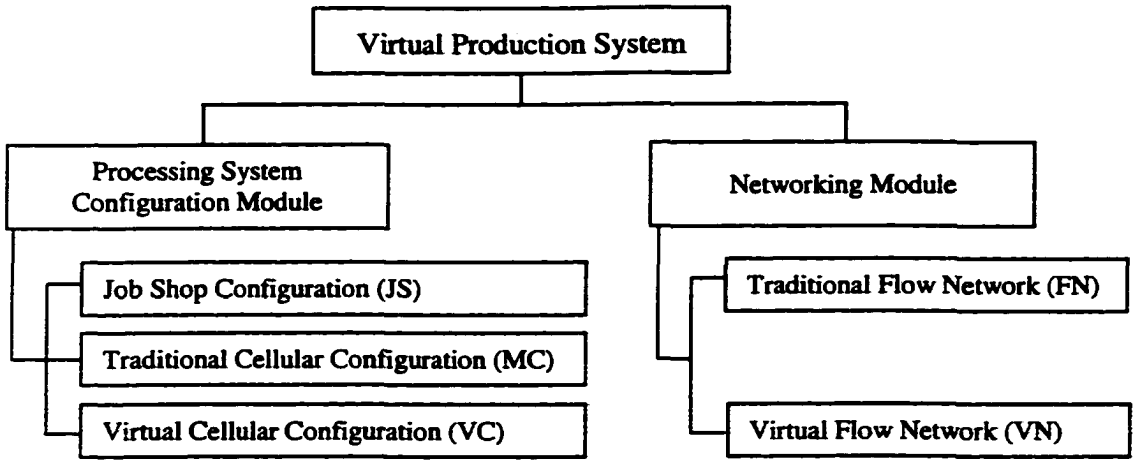


Figure 6. Virtual Production System

frequently to ensure the best performance of a shop at every production sessions. The two modules are described next.

1.2.1 The Processing System Configuration Module

For the processing system configuration module, the alternative machine arrangements are as follows:

(1) Traditional job shop (JS)

Machines are organized or grouped according to process type.

(2) Traditional cellular configuration with dispersed machine locations (MC)

Machines are grouped into fixed machine cells, except that machines in a cell do not necessarily occupy the same contiguous area. The cells are formed within an existing job shop layout and machines are assumed to be immovable. The cells do not change as the product mix changes.

(3) Virtual cellular configuration with dispersed machine location (VC)

Machines are formed into virtual cells, which are subject to change as the product mix changes and are formed specifically to respond to the product mix at hand. Machines in a virtual cell do not necessarily occupy the same contiguous shop floor area. A virtual cell is simply a logical grouping of the

machines in a computer database. Changing the grouping of the machines in the database also changes the grouping of the machines on the shop floor.

In general, all three alternatives of the processing system configuration module can be viewed as groupings of the machines in a computer database, except that the data for the traditional job shop and traditional cellular configurations are fixed and unchangeable over time.

1.2.2 The Networking Module

For the networking module, two alternative designs, the traditional flow network and virtual flow network, are considered. Furthermore, a mobile material handling system such as an AGVS that operates on a network is assumed.

(1) Traditional flow network (FN)

The flow network is also referred to in this study as the guidepath. In the traditional flow network, the flow is unidirectional in any aisle, and the direction of flow is fixed over time. The direction flow is determined once at the beginning and is never allowed to change.

(2) Virtual flow network (VN)

In a virtual flow network, the guidepath is virtual, and the direction of flow is therefore subject to change as the product mix changes. The undirected network is fixed, but the direction of flow changes as needed. In a virtual flow network, flow direction on a segment or aisle switches from one direction to the opposite as needed, provided it is possible to reach any node from any other node.

Similar to the processing system configuration module, both the traditional and the virtual flow networks can be viewed as information in a computer database, except that in the traditional network the data configuration cannot be changed, whereas in the virtual network the data can be changed to respond to a changing product mix. Figure 6 summarizes the composition of the modules.

1.2.3 Production System Configurations

Combining the three alternatives in the processing system configuration module and the two alternatives in the networking module yields six possible production system configurations. As shown in Table 2, a job shop configuration with a fixed network and a traditional cellular configuration with a fixed network are denoted as "Traditional Job Shop" and "Traditional Cellular System", respectively. The following four systems having at least one "virtual" element (VC, VN, or both) are considered as the four different editions of "Virtual Production System" and are described in the next sections. In the bottom of Table 2, three benchmark production systems, where movable machines and fixed networks are employed, are presented. The three benchmark systems are discussed in Chapter 7.

Virtual Production System Type I (VC/FN)

In the first type, the virtual cellular configuration with the traditional flow network, machines in a shop are logically organized as virtual cells in a computer database; however, the associated AGV guidepath network, once given, is fixed. Virtual cells change as the product mix changes. To form virtual cells, the Ko's virtual cell formation procedure presented in Chapter 3 can be applied.

Table 2. The combinations of production system

		Processing System Configurations		
		JS	MC	VC
Networking	FN	Traditional Job Shop[1]	Traditional Cellular System[2]	Virtual Production System (Type I)[3]
	VN	Virtual Production System (Type II)[4]	Virtual Production System (Type III)[5]	Virtual Production System (Type IV)[6]
Benchmark		Job Shop[7]	Boctor's[8]	Ko's[9]

Keys:

1-6: Fixed machines are employed

7-8: Movable machines and fixed networks are employed

Virtual Production System Type II (JS/VN)

The second type combines the job shop configuration with the virtual AGV guidepath network. The job shop configuration views a shop as a regular job shop in which machines are organized by processes and fixed over time. However, the associated virtual AGV guidepath network may be updated in each production session. The proposed AGV guidepath network design procedure presented in Chapter 4 can be used to update the associated flow network.

Virtual Production System Type III (MC/VN)

The third type combines the traditional cellular configuration with the virtual AGV guidepath network. That is, machines in a shop are conceptually grouped into machine cells by using one of the available techniques in the literature. However, machine cells once created do not change as the product mix changes. Furthermore, since the number and types of machines are fixed in a shop, machine duplication is not considered; thus, intercell movements are allowed. The associated virtual AGV guidepath network may be updated as the product mix changes by using the proposed AGV guidepath network design procedure presented in Chapter 4.

Virtual Production System Type IV (VC/VN)

The fourth type, the virtual cellular configuration with the virtual AGV guidepath network, is the ultimate virtual production system. Both the machine configuration and network design can be updated as the product mix changes. Virtual cells and AGV guidepath networks are updated by using the Ko's virtual cell formation procedure (Chapter 3) and AGV guidepath network design procedure (Chapter 4), respectively.

Each production system exhibits the characteristics of its components. This research is an attempt to develop a procedure for determining the best production system configuration to use in processing any given set of jobs released for production in the shop, with the objective of minimizing production cost or time.

It must also be pointed out that the traditional job shop configuration, traditional cellular configuration, and traditional flow network are all assumed to be determined using some initial product data assumed to exist at time zero, or the start of the process.

Moreover, the three production systems with movable machines and fixed flow networks are also investigated in the study. The three configurations (JS, MC, and VC) are still employed to physically rearrange machines in a shop. The performance of a shop with movable machines is considered as a benchmark for evaluating the performance of a virtual production system.

1.3 The Objective and Structure of the Research

The primary objective of the research is to develop a systematic procedure to select the best configuration of a production system for a given product mix, given that the base layout for the shop is a process or job shop layout. The scenario considered in this study is as follows: Given a set of jobs to be processed, the machine or process routings for the jobs, an existing process layout of the shop, and an undirected traffic flow network, develop a procedure that would determine the best production system configuration that results in the smallest sum of setup and material handling time.

As previously stated, a production system consists of the processing system configuration module and the networking module. Within these two modules, six types of production systems were identified in Table 2 (not include three benchmark systems). The procedures for designing the two traditional production systems have been well addressed in the literature. A design procedure for the four types of virtual production system has not been reported. Therefore, the task in this study is to develop a systematic procedure to configure such production systems.

Once a procedure to configure all the alternative production systems is obtained, the next objective is to compare and analyze the performance of the different production systems under some given production scenarios, to determine the overall long term effectiveness of each system relative to the others. The structure of the modeling required consists of two major parts, namely, virtual cell formation and virtual networking, which are described in detail in the following sections.

1.3.1 Virtual Cell Formation

Given a set of jobs to be completed, the first research goal is to create virtual cells. Because formation of virtual cells is based on the product mix at hand, the virtual cells are always updated and adapted to the production instance. Although many cell formation methods have been discussed and presented in the literature, few are appropriate for virtual cell formation. For instance, in most traditional cell formation approaches, a machine is restricted to serve one cell only, and intercell movements are either not preferred or not allowed. However, virtual cells might share the same machine, and products (or product families) might share the same virtual cell.

Furthermore, to apply cell formation methods presented in the literature, users are always required to provide at least one of the following parameters: the number of cells, the cell size, or the ranges of these parameters. Without these parameters, the cell formation procedures that have appeared in the literature might be unable to generate the desired cells. This situation leads users to try many different values of the parameters in an attempt to obtain a better cell configuration. However, because virtual cells are updated frequently in a dynamic environment, it seems inappropriate to configure the cells through a trial-and-error process. Besides, if a relative number of cells or cell sizes were specified, then the generated cell configuration would be limited by the specified value.

The ideal approach for virtual cell formation should be such that the number of cells and the size of a cell are naturally determined by the operation sequences of the products within a production session or product mix. In view of this requirement, development of a virtual cell formation method is necessary. The Ko's virtual cell formation algorithm is described in Chapter 3.

1.3.2 Virtual Networking

The goal of the second part of the research is to develop a systematic approach to designing the virtual AGV guidepath network. The reason that it is called "virtual" is that there are no physical taps or wires on the ground to guide AGVs. AGVs could be guided by radio or laser beams. An associated flow network exists as a database type in a computer and

AGVs follow the flow network to link machines together. A virtual AGV guidepath network changes once the product mix changes.

Several guidepath network design methods have been discussed in the literature, from mathematical approaches [22-25] to heuristic approaches [27, 28]. A proposed mathematical model with a modified Branch and Bound method [24] was claimed to yield good network solutions. However, the modified technique was a partial search method and therefore cannot guarantee the optimality of the network obtained. For a larger problem, even mathematical programming might be unable to generate a solution within a reasonable time; it can be very time consuming to formulate and solve such a mathematical model. When the requirement of quickly and frequently redesigning the transport network is considered, as in the case of a changing product mix environment, the use of mathematical programming is inefficient.

On the other hand, although heuristic approaches can generate solutions very quickly, optimal solutions cannot be guaranteed because a heuristic approach is usually developed on the basis of intuition, insight, and experience. Without being carefully examined and discussed, a heuristic algorithm might even fail to produce feasible solutions for some cases, even though a feasible solution exists, as reported in the literature [28].

In this research, in view of the frequency at which the network design may be updated, the use of a heuristic procedure rather than an exact procedure is recommended in spite of the inability to guarantee optimality; a method that can quickly generate an acceptable network solution that is near optimal is preferred. It is thus necessary to develop a more robust heuristic approach for the AGV guidepath network design problem in this work.

1.4 Research Assumptions

In the design and operation of a virtual production system, the following assumptions are made:

(1) The product types and desired volume to be produced change over time. However, during any given period or instance of job release, the production volume as well as the machine routings for the jobs are known.

(2) A base job shop or process layout is given, and the machines in the shop are not movable.

(3) There is only one unit of each type of machine in the shop, and all machines work perfectly.

(4) Constraining of production to be completed within a time interval is not considered in the research. Thus, the issue of machine capacity is not considered.

(5) An undirected AGV guidepath network exists and it is eventually converted into a unidirectional network.

(6) The pick-up and drop-off points for machines are given, and they are not located on any network intersection, so as to avoid traffic congestion.

(7) The comparison between job shop, cellular manufacturing, and virtual production systems are based on the same resource level; that is, the number and type of machines are the same among the alternatives. Therefore, for the traditional cellular manufacturing system, intercell movements are allowed in this study if necessary.

1.5 Research Tasks

The following tasks are performed in the research.

(1) Develop a virtual cell formation algorithm such that virtual cells can be reformed and updated as the product mix changes.

(2) Develop a robust AGV guidepath design algorithm with the objective of minimizing total material handling time.

(3) Implement the proposed algorithms on a UNIX operating system by coding them in C language.

(4) Analyze the effectiveness of the algorithms.

(5) Construct virtual production systems with the two proposed algorithms.

(6) Expand the study by allowing free ranging machines in a shop and develop an appropriate procedure for machine allocations.

(7) Compare and analyze the performance virtual production systems with other production systems in a changing product mix environment by using three measures: total setup time, total material handling distance, and weighted performance value.

1.6 Benefits of the Research

The expected benefits from the research are as follows:

(1) Virtual production system provides an feasible solution for a shop with fixed machines to improve its performance in a changing product mix environment.

(2) A systematic technique is provided for transforming the production type of a shop into virtual production system modes.

(3) Through the processing system configuration module, a shop might enjoy the best production configuration in terms of the lowest setup time possible under a given production scenario.

(4) Through the networking module, the material handling travel distance might be minimized under a given production scenario.

(5) Without specifying any artificial parameters, the Ko's virtual cell formation procedure can readily reconfigure machine cells in response to the changes in product mix. By involving the sharing concept, the Ko's virtual cell formation procedure can reduce the machine duplication problem and improve the intercell movement problem in traditional cellular manufacturing.

(6) The AGV guidepath network design procedure can produce a feasible and good quality network in a reasonable time and therefore can be updated easily in response to a changing product mix environment.

(7) The performance comparisons are made among virtual production systems, traditional production systems, and production systems with movable machines. Based on the results shown in the thesis, a shop could identify the production system type that is suitable for its needs.

1.7 Organization of the Dissertation

The remainder of this thesis is organized as follows: In Chapter 2, the relevant literature on traditional cellular manufacturing, virtual cellular manufacturing, and the guidepath design for the AGV system are reviewed. The Ko's virtual cell formation algorithm is described and demonstrated in Chapter 3. The AGV guidepath network design procedure is

presented and examined in Chapter 4. In addition, an example of virtual cellular manufacturing is presented in Chapter 5. Some virtual production systems are constructed and compared with the job shop and traditional cellular manufacturing systems under conditions of a changing product mix environment in Chapter 6. The study is expanded by allowing free ranging machines and observations are made among different production systems in Chapter 7. Finally, the works of the study are summarized in Chapter 8.

CHAPTER 2. LITERATURE REVIEW

As stated in Chapter 1, the research subject of this study is composed of multiple components of production system types and AGV guidepath types, and builds upon the foundations of those components. This chapter reviews some of the earlier work related to the current research and points out differences between the current research and studies reported earlier.

2.1 Cellular Manufacturing

A manufacturing system based on the philosophy of group technology (GT) is called cellular manufacturing [29, 30]. In cellular manufacturing, the parts to be produced are grouped into part families based on characteristics such as their design, process, geometric shape, and required fixtures and set ups. In addition, the machines required to produce a part family are located together to form a cell. A part family is expected to be processed entirely within its own dedicated cell.

Many benefits of the use of cellular manufacturing have been reported, such as shorter setup times, shorter lead times, reduced work-in-process inventories, and improved product quality [31]. The problem of determining part families and machine cells is called the cell formation problem (CFP) [30]. Several approaches to solving the CFP have been presented in the literature. Basically, these methods can be classified in five categories, as shown in Figure 7 [32]. The techniques related to each category are reviewed and discussed in the following sections.

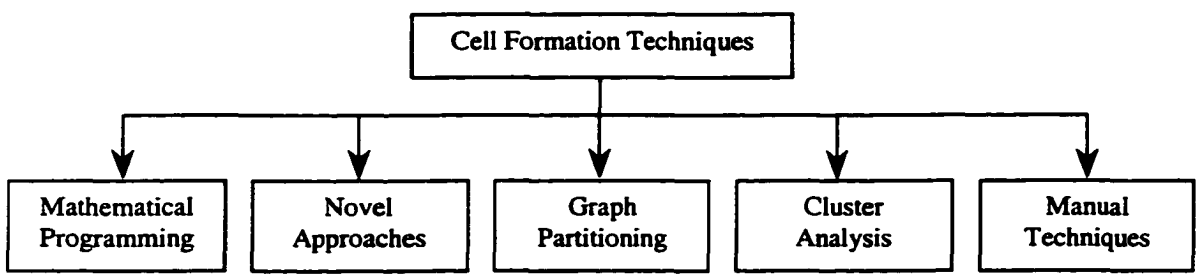


Figure 7. The classification of the CFP approaches.

2.1.1 Manual Techniques

Manual techniques require the analyst to make a series of judgments during the cell formation procedure such that part families and manufacturing cells are iteratively established by use of these manual approaches. Several manual techniques presented in the literature are reviewed as follows.

In Production Flow Analysis (PFA), first developed by Burbidge in 1963 [29, 33-35], the routing information of parts is used to simultaneously identify part families and their corresponding manufacturing cells [36]. Basically, PFA consists of three sequential steps:

(1) Factory Flow Analysis is employed to partition the problem and to obtain the simplest possible material flow system, i.e., both major groups of department size and major families of parts are analyzed and identified by using this analysis.

(2) Group Analysis is used to identify manufacturing cells so that the entire part family is completed within a cell.

(3) Line Analysis is applied to arrange machines within the same cell in order to optimize the material flow within a cell as nearly as possible.

In 1972, El-Essawy and Torrance proposed another cell formation procedure, Component Flow Analysis (CFA), in which three major steps are involved [37]. First, parts are sorted into groups based on their manufacturing requirements. Second, the groups are manually analyzed to generate manufacturing cells. Third, a detailed flow analysis is performed and appropriate adjustments are made to obtain an acceptable design. Although CFA and PFA are very similar, PFA first partitions the problem, whereas CFA does not [38].

Various other manual procedures have been reported in case studies. For instance, the Langston Company Division of the Harris-Intertype Corporation groups parts into families by visual examination. When Polaroid pictures of approximately 21,000 parts were inspected, over 93% of the parts could be classified into five families by using this technique [39]. Other study cases are available in the literature [40, 41].

Two major drawbacks of manual techniques should be addressed. First, because these techniques heavily depend on human judgement, they do not lend themselves to being implemented on a computer [37]. Furthermore, the premise underlying use of manual

techniques is that the various parts should be clearly defined, and this makes these techniques very difficult to apply [42].

2.1.2 Mathematical Programming

Several mathematical approaches have been used to identify part families and their corresponding manufacturing cells. Among them, the most basic and most popular are linear programming, non-linear programming, integer programming, and mixed integer programming [38, 43].

The CFP is usually formulated by incorporating an objective function and a set of constraints in a precise format. A variety of objectives that have been presented in the literature have been summarized [43]. Among the objectives, the six used most often are:

(1) Minimizing the number of exceptional parts [44, 45]

(2) Minimizing machine setup times [46, 47]

(3) Minimizing the inter-cell movements or the material handling cost [48-50]

(4) Minimizing investments in new machines [51-53]

(5) Maximizing the sum of similarities [54, 55]

(6) Maximizing the machine utilization levels [56-58].

These objectives could either be applied individually or be combined for a specific mathematical model. When several objectives are consolidated into one objective function, the mathematical programming is goal programming [47, 59].

Another key component of using mathematical programming is defining the system constraints, which can be classified into five types: logical constraint, which ensures that each machine and parts will be assigned to only one cell; cell size constraint, which guarantees that the number of machines in a cell will not exceed a specified upper bound; physical constraint, which includes limitations of factors such as budget, space, capacity; modeling constraint, which provides additional required connections among decision variables, objective functions, and parameters; and intuitive constraint, which confirms the non-negative and integrality properties of decision variables.

As an example of mathematical programming, a model [44] is presented and illustrated as follows:

$$\text{MIN} \quad Z = \sum_{i=1}^{M} \sum_{j=1}^{P} \frac{a_{ij}\left(\sum_{k=1}^{C} \left| x_{ik} - y_{jk} \right| \right)}{2}$$

Subject to :

$$\sum_{k=1}^{C} x_{ik} = 1 \qquad i = 1 \sim M \qquad (1)$$

$$\sum_{k=1}^{C} y_{jk} = 1 \qquad j = 1 \sim P \qquad (2)$$

$$\sum_{i=1}^{M} x_{ik} \leq m \qquad k = 1 \sim C \qquad (3)$$

$$x_{ik} \in \{0, 1\} \qquad \forall i, k \qquad (4)$$

$$y_{jk} \in \{0, 1\} \qquad \forall j, k \qquad (5)$$

where,

i = machine index;

j = part index;

k = cell index;

M = number of machines;

P = total number of parts;

C = number of manufacturing cells;

m = maximum number of machines allowed in a cell;

a_{ij} = volume of part j required to be processed on machine i;

x_{ik} = binary variable indicating whether or not machine i is assigned to cell k;

y_{jk} = binary variable indicating whether or not part j is assigned to cell k.

The objective of this mathematical model is to minimize the number of exceptional parts. Exceptional parts are those parts that must visit more than one cell to be completed. Constraints (1) and (2) ensure that each machine or part is assigned to one and only one cell. Constraint (3) ensures that the upper bound of the cell size holds; the bound is m in the example. Constraints (4) and (5) guarantee the required integrality and non-negative properties. Note here that the number of cells needs to be specified by users, which is C in the model.

The mathematical model developed for the CFP is a variation of an assignment optimization problem that is known to be NP-hard [60]. To solve the mathematical models, algorithms such as branch and bound, branch and cut (for small-size problems), and heuristic approaches are suggested [38, 61].

Several weaknesses of mathematical programming for the CFP should be mentioned. First, to develop a mathematical model successfully, either the cell size or the number of cells should be previously assigned by users. However, it is very difficult for one to know a priori how many cells are needed. Second, with regard to number of decision variables and constraints, it is very time consuming to formulate and solve a mathematical model, even for a small-size problem. Third, mathematical programming is suitable only for a stable environment; once the mix of products is changed, a mathematical model needs to be reformulated. Furthermore, for a large-size problem, it is almost impossible to find an optimal solution within a reasonable time period, a phenomenon known as NP-complete or NP-hard [43]. Therefore, mathematical programming is not widely used in practice [38].

2.1.3 Graph Partitioning Methods

Several graph-partitioning methods for the CFP have been published in the literature. These techniques treat the machines and/or parts as nodes, and the material flows as arcs. The intention of these graph-partitioning approaches is to obtain disconnected subgraphs from a machine-machine or machine-part graph [32], thereby identifying manufacturing cells.

The first use of graph theory to solve the CFP was proposed by Rajagopalan and Batra in 1975 [62]. The objective is to minimize the movements of parts between machine cells [32],

using a measure called Jaccard's similarity coefficient [63, 64], which is calculated for each machine pair. The arc between a machine pair is introduced only if the similarity value of the pair is greater than a predefined threshold value. When all qualified arcs are established, the graph is completed. Next, the cliques of the graph that represent machine groups are identified, analyzed, and merged together to create manufacturing cells. The same approach, with different similarity coefficients to design primary, secondary, and tertiary cells, was proposed by De Witte [65].

In 1988, Kusiak and Chow presented three graph types that could be used for the CFP [66]: the bipartite graph, transition graph, and boundary graph. A bipartite graph consists of two node sets, one for parts and the other for machines. An arc between a machine-part pair is introduced if the part must visit the machine. The task is to determine how to cut the graph into disjoint subgraphs such that the production flow in each subgraph is maximized [66, 67]. In a transition graph, a node represents a machine, while an arc represents a part. An arc connects two nodes if there is a part that needs to be operated on by the two machines. The transition graph is useful in detecting bottleneck machines. The third type, the boundary graph, consists of a hierarchy of bipartite graphs. It is employed to determine bottleneck parts/machines, and then generate disjoint subgraphs. However, determining the bottleneck parts/machines in a graph in order to identify disjoint graphs is very complex [66]. A heuristic algorithm for solving this problem was presented by Lee *et al.* [67] and further extended by Vannelli and Kumar [69, 67].

Other approaches in this category [70] include (1) network techniques, proposed by Vohra *et al.* [71], Wu and Salvendy [72], and Lee and Garcia-Diaz [73-75]; (2) minimizing spanning tree, presented by Ng [76]; and (3) a heuristic graph partitioning approach, developed by Askin and Chiu [77].

The major drawbacks inherent to these approaches are that practical issues such as production volumes and alternate process plans are not addressed [43, 78]. Furthermore, the clique identification problem is NP-complete [75].

2.1.4 Cluster Analysis

Cluster analysis assigns objects into clusters such that individual elements within a cluster have a high degree of relationship, while the relationship between clusters is very slight. A common feature of cluster analysis is that it sequentially rearranges columns and rows of the machine/part matrix based on an index, until diagonal blocks are generated [79]. Basically, the procedures in this category could be further divided into three types: (1) array-based clustering techniques; (2) hierarchical clustering techniques; and (3) non-hierarchical clustering techniques [32].

Array-Based Clustering

Array-based clustering requires a two-dimensional machine/part incidence array. The machine/part matrix, A, consists of elements $a_{ij} = 1$ if part j needs to visit machine i; otherwise, $a_{ij} = 0$. By using the machine/part matrix, an array-based clustering procedure is employed to produce small cluster blocks along the diagonal of the matrix. The process is accomplished by performing a series of rows and columns manipulations; ideally, each a_{ij} within a cluster block is expected to have the value of 1, and all elements outside the cluster block are expected to have the value of 0. In this way, part families and machine cells are generated simultaneously. At least eight array-based algorithms are found in the literature [32]; they include:

(1) Bond Energy Analysis, BEA, by McCormick *et al.* [80]

(2) Rank Order Clustering, ROC, by King [81, 82] and King and Nakornchai [83]

(3) Modified Rank Order Clustering, MROC, by Chandrasekharan and Rajagopalan [84]

(4) Direct Clustering Analysis, DCA, by Chan and Milner [850]

(5) Occupancy Value method, by Khator and Irani [86]

(6) Cluster Identification methods by Kusiak and Chow [87] and Longradan[88]

(7) The Hamiltonian Path Heuristic by Askin *et al.* [89].

Among them, the three most popular algorithms are BEA, ROC, and DCA, which were compared and analyzed by Chu and Tsai [79]. The comparison showed that BEA is

significantly superior to the other two algorithms, whether the problems were with or without exceptional elements and bottleneck machines.

The drawbacks of these techniques are as follows. First, most approaches consider only binary routing information and ignore other important factors such as operational sequence, and machine capacity. Next, in most cases, bottleneck machines must be removed before any machine/part cluster block can be clearly identified [43]. Furthermore, to identify cluster blocks requires human judgment, and that is difficult when a problem is large. Finally, these techniques generally ignore the production volume of parts.

Hierarchical Clustering

Unlike array-based techniques, hierarchical clustering methods do not produce machine cells and part families simultaneously. Instead, hierarchical clustering techniques operate on an input data set described in terms of a similarity/dissimilarity or distance function and then create a hierarchy of clusters or partitions [70]. At each partition, it is possible to have a different number of clusters with different numbers of machines.

Hierarchical clustering approaches can be further separated into two basic types, the divisive type and the agglomerate type. Divisive type algorithms create a series of partitions until each machine/part belongs to only one cluster. In contrast, agglomerate type algorithms start with single machine/part and proceed to merge them into larger partitions until the whole data set is involved in one partition. The only divisive algorithm in the literature was presented by Stanfel [89], while most hierarchical clustering uses agglomerate type algorithms.

Hierarchical clustering techniques consist of two steps. The first step is to calculate similarity/dissimilarity coefficients for every machine (part) pair. The various similarity/dissimilarity coefficients available in the literature include:

(1) The Jaccard's similarity coefficient [63, 64]

(2) The additive similarity coefficient [91]

(3) The multiplicative similarity coefficient [91]

(4) Production volume based similarity coefficient [92, 93]

(5) Operation based similarity/dissimilarity coefficients [94-98]

(6) Capability based similarity coefficient [99]

(7) Routing based similarity coefficients [100, 101]

(8) Weighted similarity/dissimilarity coefficients [102-104]

(9) Commonality score [105]

(10) Other similarity/dissimilarity coefficients [106-108].

Among these, the Jaccard's similarity coefficient has been found to be the most popular and most suitable for analyzing the groupability of matrices [67].

The second step involves determining how to combine or merge the machine (part) pairs together. Several algorithms have been proposed for this purpose. The Single Linkage Clustering Algorithm, SLINK, was first used by McAuley [64]. The major problem of SLINK is the chaining problem; that is, two clusters may join together based on the similarity of two members, while most other members remain apart because of lack of similarity [103]. To reduce the chaining problem, Seifoddin [109] developed the Average Linkage Clustering Algorithm, ALINK, Gupta and Seifoddini [93] presented the Weight Average Linkage Clustering Algorithm, WALCA, which is also known as WALINK, Moiser [110] developed the Complete Linkage Clustering Algorithm, CLINK, and Wei and Kern proposed the Linear Cell Clustering Algorithm, LCC. The first four algorithms were compared and evaluated with respect to their chaining effect by Gupta [111], who concluded that the chaining problem is increasingly severe in order of CLINK, WALINK, ALINK, and SLINK.

Because of their flexibility with regard to incorporating manufacturing data, the techniques are better in hierarchical clustering than in array-based clustering. However, several disadvantages exist. One is that the designer must decide on an appropriate similarity for groups. If the problem is too large, other methods for storing the hierarchy are required. Another disadvantage is that most algorithms do not handle the problem of machine duplication [70]. Furthermore, the problems of how to select the cluster criteria and the performance measure and how to determine the number of clusters remain to be solved [44].

Non-Hierarchical Clustering

Non-hierarchical clustering methods are iterative approaches. Basically, a non-hierarchical clustering technique operates on an input data set by prespecifying the number of clusters to be formed using a similarity function. The input data set could be either an initial partition of the data set or the choice of a few seed points [32]. The major difference between hierarchical clustering and non-hierarchical clustering is that a similarity matrix does not need to be computed and stored in most non-hierarchical clustering algorithms [70]. However, because of the predefined number of clusters, some clusters may be forced to be consolidated or split, in order to meet the specified cluster number. Several non-hierarchical clustering techniques are reviewed in the sections that follow:

A technique named the Ideal Seed Non-Hierarchical Clustering algorithm, ISNC, was developed by Chandrasekharan and Rajagpoalan [112]. It is used in three stages. First, the CFP problem is represented as a bipartite graph, and the upper limit of the number of clusters is derived as k. A modified MacQueen's k-means method is then employed by choosing the last k data units or vectors as initial seed points [112, 113]. Second, the remaining data units are assigned to the cluster with the nearest centroid. After each assignment, the centroid is updated to include the current data unit. Third, the output of the second stage is improved in terms of both utilization and intercell movement by introducing "ideal seeds". The ideal seeds are generated for columns and used as fixed seed points for further clustering. Moreover, an evaluation criterion called "grouping efficiency", η, is used to measure the intercell movements and the machine utilization in a cell.

Another technique proposed by Chandrasekharan and Rajagpoalan, named ZODIAC (zero-one data: ideal seed algorithm for clustering) [114], is an expanded and improved version of ISNC. A new concept of "relative efficiency" was developed as a stopping rule for the iterations. Moreover, for the initial seed choice, four options are introduced: arbitrary seeds, artificial seeds, representative seeds, and natural seeds.

Srinivasan and Narendran in 1991 revealed their non-hierarchical technique GRAFICS (grouping using assignment method for initial cluster seeds) [115]. GRAFICS can generate initial seeds from an assignment problem that maximizes the similarity between machines, with a maximum density rule used as the cluster criterion.

When a clustering algorithm for sequence data, CASE, was presented recently by Nair and Narendran [104], a new similarity measure and new seeding techniques were introduced. In addition, the number of clusters to be formed is based on a threshold affinity level, TAL. At the beginning, the TAL is set equal to "0," and the similarity measure is computed for every machine pair. Any pair of machines belonging to different clusters will have the similarity measure of "0," and therefore, the first two clusters are formed relative to the two machines. The procedure is terminated if the bond efficiency of a new solution is worse than that of the existing best one. Otherwise, the TAL is updated and the procedure is continued to the next iteration. CASE differs from other non-hierarchical clustering algorithms in that the similarity matrix must be computed and stored. Furthermore, the number of clusters to be formed is controlled by the measure of bond efficiency with TAL.

Several comparisons of these techniques have been reported in the literature. For instance, that GRAFICS outperformed ZODIAC in grouping efficiency and grouping efficacy was reported in [116]. A comprehensive comparison of nine well-known algorithms, including array-based clustering techniques, hierarchical clustering techniques, and ISNC, has been reported [117]. It showed that ISNC outperformed the other eight algorithms. The major drawback of non-hierarchical clustering is related to selection of the seed. Arbitrariness in the choice of seed points could lead to unsatisfactory results [32].

2.1.5 Novel Approaches

The category of "novel approaches" consists of relatively new approaches to the CFP. The major features of these methods are the use of artificial intelligence and/or pattern recognition techniques, and search approaches to form machine cells or part families. These approaches can be further classified into six types: expert system, fuzzy logic, neural network, genetic algorithm, simulation, and other search techniques.

Expert System

Knowledge-based rules and pattern recognition techniques are the two necessary components of expert systems. Although few papers have been presented in the literature, use of the expert system for the CFP is a promising area to explore [78].

In 1986, Wu *et al.* presented an algorithm for using syntactic pattern recognition for the CFP [118]. According to Tam [119], the advantages of syntactic pattern recognition include cell formation that takes into account material flow patterns, operation precedence relations, and non-uniform importance of machines.

In 1988, Kusiak [120] introduced a knowledge-based system that takes advantage of expert system techniques and optimization in which machine capacity, material-handling capabilities, technological requirement, and cell dimensions are considered in forming cells. About the same time, ElMaraghy and Gu presented a system that considered knowledge rules and syntactic pattern recognition techniques to form part families [121]. The basic difference between these two approaches is in the degree of automation [32].

In another algorithm proposed in 1991 by Singh and Qi [122], the concept of multi-dimensional similarity coefficient using syntactic pattern recognition was introduced to form part families.

Fuzzy Logic

Although some objects obviously belong to certain clusters, in other cases it is not clear which cluster is most appropriate. Fuzzy logic techniques are used to deal with the issues of vagueness and uncertainty in the CFP.

Fuzzy approaches can be divided into two types, classical and modern [123]. The classical fuzzy clustering techniques include techniques such as fuzzy *c*-means clustering and fuzzy mathematical programming. The modern fuzzy clustering techniques include techniques such as fuzzy neural networks.

The purpose of fuzzy *c*-means clustering, which is a modification of the *k*-means clustering method, is to minimize the Euclidean distance between the data set and the relative cluster center with specified degree fuzziness. The techniques proposed in the literature include those of Chu and Hayya [124], Gindy *et al.* [125], Wen *et al.* [126], and Leem and Chen [127].

In fuzzy mathematical programming, conventional mathematical programming and fuzzy logic are married together for solving the CFP. In 1995, Tsai [128] proposed a fuzzy mathematical programming model to form machine cells and at the same time minimize the

cost of eliminating exceptional elements. Later, the proposed model was extended to a multi-objective model. Most recently, the CFP was formulated as a fuzzy mixed integer-programming model by Tsai *et al.* [129], who presented a new fuzzy operator and examined the impact of different membership functions and operators on solving the model. Other classical fuzzy clustering techniques include fuzzy single linkage clustering and fuzzy rank order clustering. In 1992, two fuzzified clustering techniques were proposed by Zhang and Wang [130].

Another type of fuzzy clustering is to combine fuzzy logic and neural networks for solving the CFP. Adaptive Resonance Theory Model, ART, is one of the most popular neural networks employed in this area. The hybrid model (Fuzzy ART), which was first developed by Burke and Kamal in 1992 [131, 132], was investigated and extended by several other workers [133, 134]. According to Venugropal [123], because fuzzy models' capabilities are not fully exploited in the work to capture the fuzziness in part features and machining processes, this is a promising area for further research.

Neural Networks

Use of neural networks to solve problems has been reported successfully in many fields [135, 136]. Neural networks can mimic the operation of neurons to learn from experience, adapt to new situations, generate decisions, and provide reliable classifications and approximations of data. Basically, neural networks could further be classified into two types, unsupervised or supervised, some models of which are reviewed as follows.

To use a supervised neural network, a training data set including a series of input/output pairs is required to train the network by adjusting the weights between the individual nodes, neurons. The network with the trained weights is then employed as the basis for classifying new inputs. The most popular technique of this type is the back propagation algorithm [137-140].

Another technique of this type is the stochastic neural network model. As has been shown, [141], the CFP is first formulated as an integer-programming model; a stochastic neural network is then used to solve the model. However, the drawback of stochastic neural

network models is that specifying the values of various parameters and weights is more complicated than in the deterministic neural network model [123].

The other type of neural network, the unsupervised neural network, is able to self-organize the presented data to discover common properties without using any classified output data. Because the manufacturing environment is dynamic, it is difficult to know the patterns of existing parts and processes, and the unsupervised neural network is therefore more appropriate than the supervised neural network for the CFP [142]. Unsupervised neural network models reported in the literature include the following:

1. Competitive Learning (CL) models [143-147]
2. Interactive Activation and Competition (IAC) models [142, 148-150]
3. Self-Organizing Map (SOM) models [151-155]
4. Adaptive Resonance Theory (ART-1) models [156-161]

To solve the CFP, a NP-hard problem, what is needed is an approach that can generate a good quality solution within a reasonable time. Application of neural networks to cell formation problems promise as one such approach [78].

Genetic Algorithm

Genetic Algorithm, GA, mimics the evolutionary process by combining the survival of the fittest among solution structures with a structured, yet randomized, information exchange and creation of offspring [162]. GA solves linear and nonlinear problems by exploring all regions of the state space and exponentially exploiting promising areas through mutation, crossover, and selection operations [163]. Actually, GA has been applied successfully in many areas [164-167]. Some of the published work using GA for the CFP is reviewed as follows.

Venugopal and Narendran in 1992 proposed a GA-based approach to solve the CFP [168]. The objectives of the model are to minimize the intercell movements and the total within-cell load variation; limitations of machine capacities, production amounts, and processing times of parts are considered in the paper. In 1995, Gupta *et al.* presented a similar GA to minimize a weighted total number of intercell and intracell movements [169].

Later, their study was extended by adding one more objective that minimizes the total within-cell load variation [170].

Most recently, Hsu and Su developed a GA approach to solving the CFP by considering transportation cost, machine investment cost, intracell machine loading imbalance, and intercell machine loading imbalance. About the same time, a solution obtained by using two chromosomes was proposed by Cheng *et al.* [172], whose objectives were to minimize intercell and intracell moves. In another paper presented by Cheng *et al.* [108], the CFP is first formulated as a travelling salesman problem (TSP), after which a GA is developed to solve the TSP.

There are two major differences between GA and traditional search algorithms. First, instead of improving a single solution, GA simultaneously examines and modifies a population that is a set of solutions. Second, GA is able to extract information from a population and then direct the search; by so doing, GA may avoid the problem of local optimal. With these two features, GA can handle even NP-hard problems successfully, which makes GA another choice for solving the CFP.

Simulation

Unlike the literature on other techniques, there are very few papers on research using simulation approaches to solve the CFP. The only such paper found, published by Kamrani *et al.* in 1998 [183], was the first report on use of mathematical programming techniques to form part families and machine cells. A simulation model was then developed to adjust the final design by incorporating other real world data into it.

Because it considers a higher degree of realism, the simulation-based model makes more pessimistic projections than other techniques do. The major drawback of using simulation techniques is the heavy computational load, such as more than 400 hours computation time reported by Kamrani *et al.*

Other Search Techniques

Because the CFP is NP-hard, it is difficult to obtain the optimal solution for a large problem within a reasonable time, which is the reason why so many heuristic search

approaches has been developed. In the literature, the two most popular general search algorithms are Simulated Annealing (SA) and Tabu search. The two techniques try to obtain the optimal or near-optimal solution based on an initial solution, which is usually generated by using mathematical models. The major drawback of these techniques is that users need to set some parameters before initiating the search. No doubt these parameters influence the quality of generated solutions. How to determine the parameters' values for different search algorithms is another interesting area of research.

SA is a heuristic method based on iterative improvement. The basic idea is to generate random displacements from any current feasible solution and to accept as a new current solution not only solutions that improve the objective function, but also some that do not improve it. The worse solution is accepted based on the probability, $\exp(-\Delta f/T)$, where $-\Delta f$ is the amount of deterioration of the objective function and T is a tunable parameter, the temperature. Several papers report using SA for solving the CFP [173-176].

In Tabu search, another common search method reported in the literature, as in SA, an initial solution is required before the search is used. Tabu search uses memory functions of various time spans, such as the short-term memory (the Tabu-list size), to intensify the search, and the long term memory to diversify the search into new regions. In this way, Tabu search is able to overcome the local optimal problem. Several papers using Tabu search have appeared in the literature [177-182].

The techniques for solving the CFP have been classified into five categories, and have been briefly described. Basically, most techniques share the same weaknesses. First, they were developed for a stable environment, and few of them address the CFP in a changing product-mix environment. Second, human judgment must be involved in some methods to determine values such as the cell size, the number of cells, the required parameters, and so forth. Third, the part operation sequence is often disregarded. In addition, the concept of machine sharing is not allowed in these traditional cellular manufacturing design techniques. However, these traditional approaches do provide some basic ideas and information for developing virtual cells.

2.2 Virtual Cellular Manufacturing

Virtual cellular manufacturing differs from cellular manufacturing in two respects: the sharing concept and the physical location of machines in a cell. With regard to these characteristics, virtual cellular manufacturing can provide the advantages of cellular manufacturing in a job shop type plant.

In a traditional cellular manufacturing system, a part family is expected to be processed entirely in its dedicated machine cell. Any movements of parts between cells are not encouraged; hence, machine duplication and utilization problems will incur. Fortunately, applying the machine-sharing concept can reduce these problems. In a virtual cellular manufacturing system, machine cells can share the same machine on a different time schedule. Therefore, the duplication cost of machines could be saved, and machine utilization could be improved.

The other concern is the physical locations of machines within a cell. In cellular manufacturing, machines must be physically reorganized whenever machine cells are reformed, to ensure that machines belonging to a cell occupy the same area or zones on the shop floor. The reason for locating machines in one zone is to minimize material handling time. Such area location is justified if the part family intended for a cell is stable over time and has sufficient production volume. Unfortunately, such joint area location is unjustified for a shop with an unstable product mix, because the need for a particular cell configuration does not last long. In some production situations with existing layouts, traditional cells may not be feasible because of the cost of machine relocation or incompatibility of processes (e.g., welding and painting) required by parts in the same family. Where traditional cells are unjustified because of changing product mix, high cost of moving and relocating machines, or the incompatibility of processes, virtual cells offer the best alternative if the benefits of cellular manufacturing are to be realized.

In virtual cellular manufacturing, machines belonging to the same cell do not necessarily occupy a given zone or area on the shop floor; cells in a virtual cellular manufacturing system exist in a logical state instead of a physical state. Thus, they can easily be reconfigured in response to changing characteristics of the job mix on the shop floor. To ensure that material handling time is reduced or controlled, machines and cells in a virtual

system are linked together by an automated material handling system through a virtual material flow path or network. In a virtual material handling flow path, the direction of material flow in an aisle is not fixed but changes in response to changing product mix. Therefore, the task of operating a virtual cellular manufacturing system is not just the formation of the virtual cells but also the design of a virtual material flow network that links machines and cells, to minimize the total material handling distance or time.

Because of these characteristics, virtual cellular manufacturing might be expected to perform better than traditional cellular manufacturing in a dynamic environment. Unlike cellular manufacturing, virtual cellular manufacturing is in its infancy. The reported work is reviewed in this section.

The virtual cell concept, first proposed by McLean *et al.* in 1982 [21], extends the concept of the traditional machine cell by allowing the time sharing of workstations with other virtual cells that produce different part families but that have overlapping resource requirements. Machine cells are no longer identifiable as a fixed physical group of machines, but dynamically regrouped in a computer. A shop based on virtual cells is believed to provide greater flexibility than traditional machine cells do. The evolution of machine cells and the required control structures were also addressed in the paper. Because McLean *et al.* were pioneers in this area, the paper tended to be conceptual and introductive. The discussion of virtual cells was more control-oriented than design-oriented.

In the papers by Drolet *et al.* [3, 22], a production system with virtual cells was for the first time called a *Virtual Cellular Manufacturing System*, VCMS. The decision elements or design factors (the variety of machine types, the number of machines of each type, and machines' physical distributions throughout the shop) were presented for use in planning virtual cellular manufacturing layout. According to the paper, simulation study suggests that VCMS performs as well as or better than traditional cellular manufacturing. However, the overall design scheme of VCMS remains to be explored.

Rheault *et al.* [181, 182] proposed a framework, *Dynamic Cellular Manufacturing System* (DCMS), to reconfigure virtual cells physically if workstations are movable. An integer-programming model was developed to obtain the optimal location of all workstations within the shop, with the objective of minimizing overall handling costs. However, the difference

between DCMS and traditional cellular manufacturing still remains to be investigated. The design issue of virtual cell was not addressed, either.

Another simulation approach to analyzing the performance of Virtual Cellular Manufacturing, VCM, was presented by Kannan and Ghosh [20]. Five different VCM configurations were employed in the investigation. According to the paper, VCM configurations physically resemble the process layout, but differ in how they allocate machines to families to form virtual cells. The authors concluded that the benefits of cellular manufacturing could be achieved by dedicating machines to families on a temporary rather than a permanent basis, and allowing cells to respond to changes in the shop. By using the same simulation model, Kannan provided further analysis and comparison [183, 184].

In 1993, Irani *et al.* [42, 185] investigated the machine-sharing problem in VCM by exploiting layout design and intercell flows. A flow-based approach for the formation of virtual manufacturing cells was developed. However, in using the proposed scheme, a shop still needs to be physically reorganized in order to configure virtual cells.

Although several papers have appeared in this area, most are control-oriented or simulation-oriented. When physically moving machines is infeasible, systematic design of a virtual cellular manufacturing system that involves the machine-sharing concept has not been discussed. Once virtual cells are realized, the next concern in use of virtual cellular manufacturing is effective design of a network to connect the required machines together. The network design issue is discussed and reviewed next.

2.3 AGV Guidepath Network Design

To apply the virtual cell concept in a shop, an appropriate material handling system is a must. AGVs (Automated Guided Vehicle) will be adopted in this research because of their many advantages, such as flexibility, reliability, and safety. Several important issues need to be addressed when AGVs are employed. These issues include the fleet size, control system, unit load specification, locations of pick-up/drop-off points, and guidepath design issues. Because a network is needed for AGVs to link up machines, the research will concentrate on the unidirectional AGV guidepath network design issue.

Several unidirectional guidepath design techniques have been discussed in the literature. The approaches employed can be grouped into two categories. The first is the mathematical programming technique, while the other consists of heuristic approaches.

2.3.1 Mathematical Programming

The first mathematical model formulated for the AGV guidepath problem was presented by Gaskins and Tanchoco in 1987 [25]. The problem is formulated as an integer-programming model in which the nodes represent pick-up or drop-off points and aisle intersections, and the arcs represent the paths between the nodes. The objective of the mathematical model was to find an optimal unidirectional network such that the total flow distance is minimized. Moreover, two types of constraints are given. The first is a connectivity constraint used to ensure that a node will have at least one entering arc and one leaving arc. The other is a reachability constraint to ensure that each node is reachable from any other node.

The key issue in using mathematical programming is how to formulate all feasible paths between nodes in the objective function and constraints. In practice, the number of feasible paths for each pick-up/drop-off pair will increases greatly when the number of machines increases. Therefore, for a large size problem, it is hard to consider all possible paths in a mathematical formulation. In view of this, Gaskins and Tanchoco considered only the 4 shortest paths for each node pair in their model: the clockwise and counter-clockwise considerations for the pick-up point and the drop-off point, respectively. Thus, the mathematical model was able to solve the AGV guidepath design problem. However, the reduction of feasible paths renders the proposed model unable to guarantee the optimum network design. For a larger problem, it is highly possible that even a feasible network could not be obtained because none of the 4 paths are available for some node pairs. Furthermore, solving the mathematical model was very time consuming. These drawbacks make the approach unsuitable for solving larger size problems [28].

Goetz and Egbelu modified Gaskins and Tanchoco's integer-programming model by considering only major flows [23]. Because of this consideration, the number of constraints in the modified mathematical model is considerably reduced. However, the issue in their

model became one of determining which flows constitute the major flows. In addition, like its predecessor, the proposed model cannot assure finding the optimal solution, and it is valid only for small size problems.

In 1990, Kaspi and Tanchoco proposed an alternative integer-programming model for the AGV guidepath design problem [24]. A modified branch-and-bound approach with depth-search first and backtracking was employed to obtain the optimal solution. The proposed search procedure explores a pair of arcs at a time and fixes the direction of the arc that yields the lower objective value; then, the procedure branches and investigates another pair of arcs, until the best result is obtained or no more branches are available. However, because the proposed algorithm will not evaluate all the undirected arcs at the same level, it is not an exhaustive search method, and the optimality of the generated solution therefore could not be guaranteed. Moreover, to solve the problem by using the proposed branch and bound technique was very time consuming.

Another branch and bound algorithm was presented by Venkataramanan and Wilson in 1991 [26]. Initially, the shortest paths are found for each pick-up and drop-off point pair. If any conflict exists, that is, if an arc has different directions in two shortest paths, the objective values will be evaluated for each direction. Then, the searching tree will be branched from the node with the lower objective value. However, the test problems used in the paper are not generalized. The test problems can be easily separated into two groups, one having all pick-up points and the other having all drop-off points. Therefore, to validate the proposed approach, more generalized examples need to be examined. Furthermore, the computational load needs to be addressed. Because they are exhaustive searching algorithms, branch and bound techniques are suitable only for small-size problems [28].

The formulation of the AGV guidepath design problem has been discussed in the literature. As has been described, each proposed mathematical approach has major defects. Moreover, how to solve the formulated mathematical model is an issue to be discussed. Usually, the computational load is the main concern. As a result, the use of heuristics as an alternative solution approach was undertaken.

2.3.2. Heuristic Approaches

While several mathematical models have appeared in the literature, Kouvelis *et al.* proposed five heuristic unidirectional guidepath design approaches in 1992 [28], claiming that a solution could be generated in a very short time by using their heuristic methods. However, the solutions obtained were not highly accurate. Further, the presented heuristic procedures do not guarantee finding feasible solutions to problems, even if feasibility can be demonstrated.

In 1995, Seo and Egbelu presented a paper in which the AGV guidepath design problem was formulated as a mixed integer-programming model and then solved by the branch and bound technique and a heuristic algorithm [27]. In their model, the concept of flow link was introduced. While a path is defined as a physical route from a pick-up point to a drop-off point, a flow link is a logical connection between the node pair that has material flow exchange. According to the paper, the branch and bound method will produce a partial AGV network and the heuristic algorithm will then complete the network design.

Another heuristic approach for solving the AGV guidepath design problem was presented by Sugiyama *et al.* [188]. An integrated technique employing Genetic Algorithm (GA) and Simulated Annealing (SA) was proposed, and good performance was reported for it. However, considering the computation time required, the proposed approach is efficient for only small and medium size problems [188].

Using heuristic approaches might generate an AGV network very quickly; however, the quality of solutions obtained is not very good. Mathematical programming, on the other hand, could produce better quality solutions than heuristic algorithms, but the computational load is the major concern. Therefore, a robust heuristic algorithm based on mathematical considerations is preferred in this study.

Although some work has been reported in the literature, the virtual cell formation procedure and a robust AGV guidepath network design procedure need to be developed. In the next chapter, the proposed procedures for virtual cell formation and AGV guidepath network design are presented.

CHAPTER 3. VIRTUAL CELL FORMATION

The scenario considered in this study has the following known characteristics: a shop layout with an undirected AGV guidepath network, a set of jobs with known machine/process routings, and the desired demand for each job. The primary objective is to apply the available data to construct a virtual production system. A virtual production system consists of two modules, namely, the processing system configuration module and the networking module. The processing system configuration module is focused on identifying the best way to organize the machines to obtain the lowest total machine setup time. The process or machine configurations considered are the job shop configuration, the traditional cellular configuration, and the virtual cellular configuration. Of the three process configurations, the procedure for constructing or forming virtual cells has not appeared in the literature and therefore needs to be developed. The Ko's virtual cell formation procedure is presented in this chapter.

Just as the process system configuration module is focused on process layout, the networking module is concerned with the design of the AGV guidepath. The development of a procedure to construct the AGV guidepath, traditional or virtual, is presented in Chapter 4.

3.1 Introduction

The major differences between a traditional cell and a virtual cell are in the concept of machine sharing and the non-permanency of the cell configuration. That is, multiple virtual cells can share the same machine, and virtual cells are reconfigured in response to the product mix. In this study, the sharing concept is also applied to parts or part families as well. In other words, a virtual cell can serve multiple parts or part families if necessary. With the concepts of sharing in mind, the Ko's virtual cell formation procedure is presented.

The chapter is organized as follows: Fundamental concepts relevant to the developments in the chapter are briefly described in Sections 3.2 - 3.4. Next, the required input data, terminology, and related techniques are presented in Sections 3.5 and 3.6. The Ko's virtual cell formation procedure is presented in Section 3.7. Finally, test results and discussion of the results are given in Sections 3.8 and 3.9, respectively.

3.2 Machine Pattern

In this study, the machine-sharing concept is applied to virtual cells, so that a virtual cell could serve multiple parts or part families if necessary. Actually, the cell-sharing concept has its roots in machine patterns. When one looks through the operational sequences of parts, some machine patterns might exist. That is, a machine pattern consisting of at least two machines could be used to produce multiple jobs. Figure 8 shows an example of the routings for three jobs. In the figure, the numbers in the upper part of the blocks represent workstation identities, while the identities of machine patterns are represented in the lower part of the blocks. As shown in the figure, to finish job 1, eleven machines are required, machines 2, 5, 8, 11, 12, 3, 6, 1, 4, 7, and 9, visited in that sequence. When one carefully examines the three routing sequences, some machine patterns do emerge. For example, job 1 and job 3 visit the same machine cluster/pattern, *pattern 3*, consisting of machines 1, 4, 7, and 9, for production. Furthermore, using the concept of machine patterns, the operational sequences of jobs could equally be represented in pattern format. For instance, to produce job 1, three machine patterns, *pattern 1*, *pattern 2*, and *pattern 3*, should be visited in that order.

From another perspective, a machine pattern is a compact cell. To complete a job, such compact cells may be linked together in a specified order. On the one hand, the process can

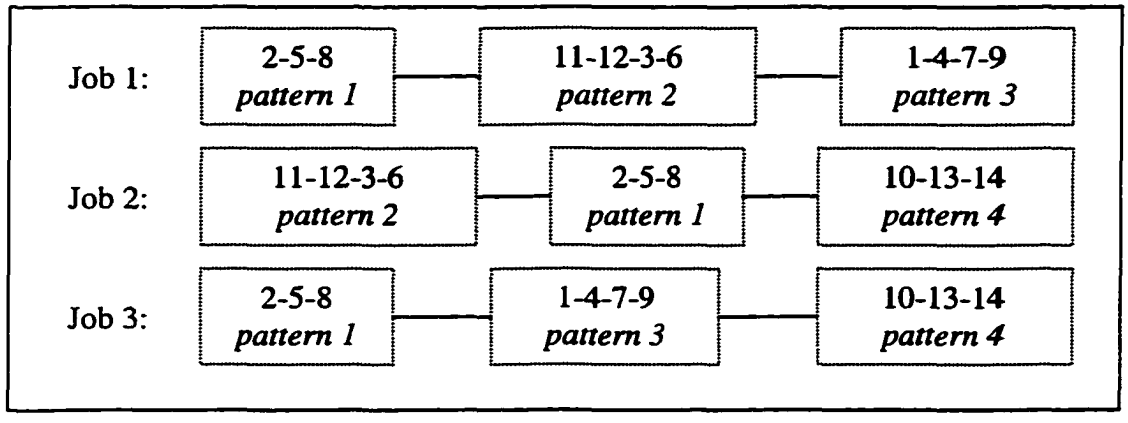


Figure 8. An example of machine routings

be seen as the separation of a traditional cell into several compact cells. On the other hand, a traditional cell can be viewed as consisting of several sequential compact cells. A series of linked compact cells is very similar to a train, each bogie of which has its own function and capacity. A train just needs to link the correct number of bogies required to satisfy customers' demands. The same idea applies to the work undertaken in this research.

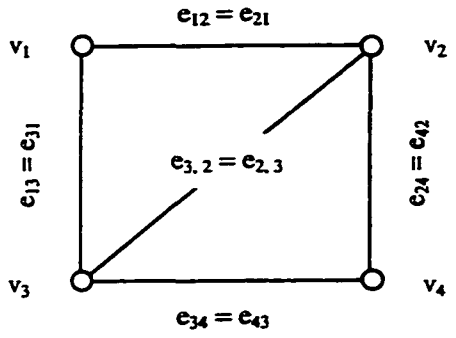
The advantages of compact cells are as follows. First, a compact cell can be used to produce multiple jobs/products, thereby reducing the problem of duplicating machines, as is done in a traditional cellular manufacturing system. Second, because a compact cell is smaller than a traditional cell, unnecessary intra-cell movements are decreased. Third, compact cells are linked according to the operational sequence of a product, thus decreasing the routing problem encountered in traditional cellular manufacturing. Furthermore, because a compact cell is a cell, the advantages of a traditional cell are still believed to hold.

The concepts of sharing machines and sharing cells will be integrated in this work. The definition of virtual cellular manufacturing is modified from Drolet [22] in that not only can a machine serve multiple cells, but a virtual cell can also serve several part families (or parts). In other words, machine cells can share the same machine, and virtual cells can be linked together to produce a product in a specified sequence.

3.3 Graph Theory

A graph G(V, E) consists of a set of vertices, V, and a set of edges, E. The vertices in a graph represent points, while the edges represent line segments connecting pairs of vertices. A vertex i is represented as v_i, and the edges between v_i and v_j is represented as e_{ij} or e_{ji}. In Figure 9, there are 4 vertices and 10 edges, where V(G) and E(G) represent the vertex set and the edge set in the graph, G, respectively. When edges are assigned directions as shown in Figure 10, they are called arcs. The direction of an arc points in the direction of the arrow. An arc pointing from v_i to v_j is denoted as Arc $_{i,j}$. For example, the arc pointing from v_2 to v_1 in Figure 10 is represented as Arc $_{2,1}$.

The number of edges connecting to a vertex, v_i, is called the in-degree of v_i, while the number of edges connecting from the vertex is called the out-degree of v_i. Obviously, the in-degree is equal to the out-degree for any vertex in an undirected graph where the directions



$V(G)$: $\{v_1, v_2, v_3, v_4\}$

$E(G)$: $\{e_{12}, e_{21}, e_{13}, e_{31}, e_{32}, e_{23}, e_{24}, e_{42}, e_{34}, e_{43}\}$

Figure 9: An example of a graph with vertices and edges

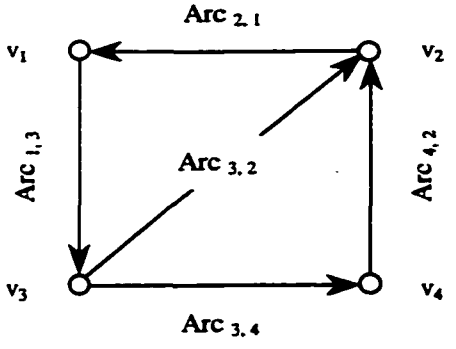


Figure 10: An example of a directed graph

of all edges are not specified. For instance, as shown in Figure 9, three edges (e_{12}, e_{32}, and e_{42}) connect to v_2 and three edges (e_{21}, e_{23}, and e_{24}) connect from v_2. Therefore, the in-degree of v_2 is 3, and the out-degree of the vertex is also 3.

However, in a directed graph in which the directions of edges are assigned as shown in Figure 10, the two degrees are not necessarily equal. For example, there is only one arc leading away from v_2, so the out-degree of v_2 is 1, but there are two arcs leading to v_2, so the in-degree of v_2 is 2. The concepts of in-degree and out-degree are employed in the study to indicate relationships between machines.

3.4 Set Theory

A set might include many elements or might be empty. In this research, elements represent machines, and a set includes several machines. A virtual cell could be extracted from one set or from two or more sets. Several related concepts in set theory are employed in the development of the Ko's virtual cell formation procedure.

Subset

If a set, A, includes all elements in another set, B, then set B is called a subset of set A and is represented as $B \subseteq A$. An illustration of a subset is shown in Figure 11.

Union

Two sets could be united or merged together to form a new set. The union of set A and set B is represented by $A \cup B$. As shown in Figure 12, set C is the union of set A and set B.

Intersection

The intersection of sets is the elements shared by the sets at the same time. The intersection of two sets is represented by $A \cap B$. An example is shown in Figure 13.

3.5 Input Data

The required input data of the Ko's virtual cell formation procedure is a set of jobs in which the machine/process routing and the production volume for each job are known.

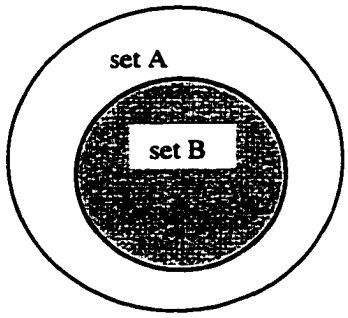


Figure 11. An example of Subset

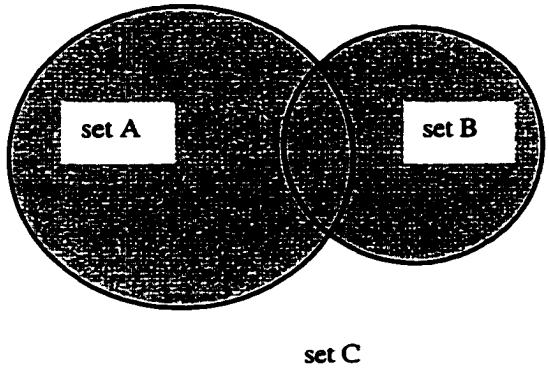


Figure 12. An example of Union

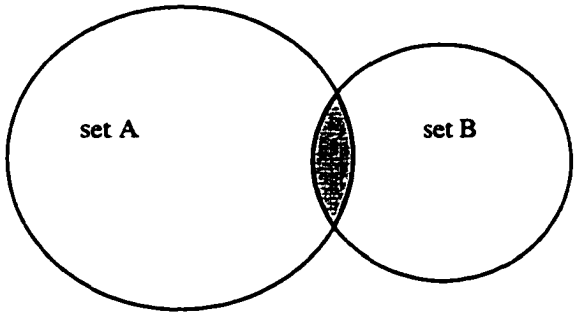


Figure 13. An example of Intersection

Consider the production situation shown in Figure 14 [61], in which 4 parts are to be produced. The machine routings and desired demand of parts are also described in the input data. For example, Figure 14 shows that part 2 must sequentially visit 12 machines (machines 11, 10, 12, 7, 13, 14, 15, 16, 17, 1, 2, and 3) to be finished, and the desired demands is 150 units. Using the data of Figure 14 as a production scenario, the layout of the facility as well as the locations of the machines are represented in Figure 15.

3.6 Terminology and Techniques

The terminology and the techniques used in the study are defined and presented in this section.

3.6.1 Dummy Machine

Machine 0 is used in this research as a dummy machine, which is not a real machine but which is used to represent the beginning and the end of a machine routing for a part. Therefore, two dummy machines will be employed to expand the machine routings of jobs in the first and last positions. For example, after being expanded by the two dummy machines, the machine routing for part 2 is modified to **0**-11-10-12-7-13-14-15-16-17-1-2-3-**0**, as shown in Figure 16. Machine 0 is also used to represent candidate cells in the part routings.

Part 1: 9-7-8-5-4-18-5-6-10-1-2-3	Demand: 200 units
Part 2: 11-10-12-7-13-14-15-16-17-1-2-3	Demand: 150 units
Part 3: 9-7-8-5-11-10-12-7-13-14-15-16-17	Demand: 325 units
Part 4: 9-7-8-5-4-18-5-6-10-13-14-15-16-17-1-2-3	Demand: 405 units

Figure 14. An example of a set of jobs with machine/process routings

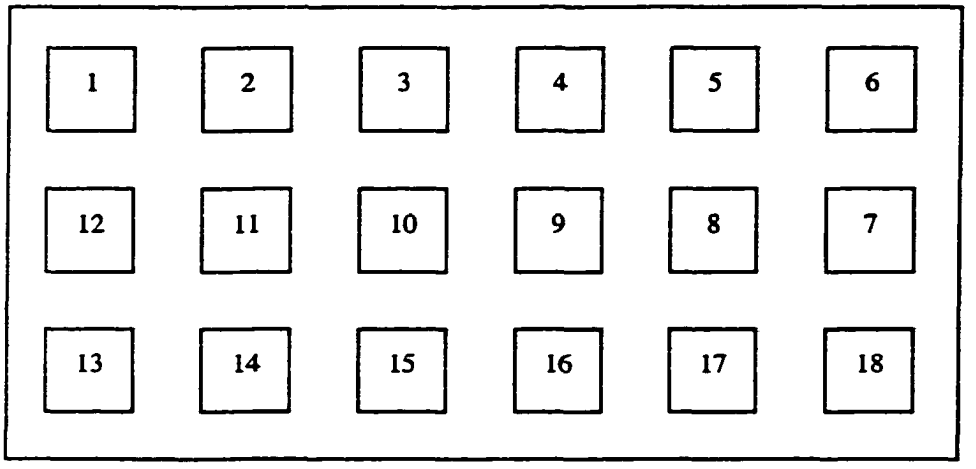


Figure 15. An illustrative shop layout with 18 machines as given in Figure 13

Part 1: **0**-9-7-8-5-4-18-5-6-10-1-2-3-**0**	Demand: 200 units
Part 2: **0**-11-10-12-7-13-14-15-16-17-1-2-3-**0**	Demand: 150 units
Part 3: **0**-9-7-8-5-11-10-12-7-13-14-15-16-17-**0**	Demand: 325 units
Part 4: **0**-9-7-8-5-4-18-5-6-10-13-14-15-16-17-1-2-3-**0**	Demand: 405 units

Figure 16. An example of a set of jobs with dummy machines

For instance, if a candidate cell consists of machine 12 and machine 7, then the routing of part 2 is modified to 0-11-10-0-13-14-15-16-17-1-2-3-0. Note that, the original positions of machines 12 and 7 are updated to one dummy machine only. The definition and method of creating candidate cell are given later in this section.

3.6.2 Exceptional Position Machine

An exceptional position machine is a machine that is directly bounded by two dummy machines in the routing for a part or job. During the application of the Ko's virtual cell formation procedure, candidate cells might be produced. The part routings will then be updated by dummy machines to represent these candidate cells. Hence, when a machine is isolated by two dummy machines in a part's routing, the machine is considered to be an exceptional position machine. For example, consider an updated part routing, 0-11-10-0-15-0-16-17-0; because it is bounded by two dummy machines, machine 15 is an exceptional position machine in the job routing.

3.6.3 From-To Table

In a From-To Table, the input data is represented by machine pairs and listed sequentially. The "From-To" means the link order between the machines in a machine pair. For example, the From-To Table for the jobs shown in Figure 16 is represented in Table 3. To produce part 1, the first machine visited is the dummy machine, machine 0, and the second machine visited is machine 9. Therefore, in Table 3, machine 0 is in the "From" column, and machine 9 is in the "To" column. In this way, the input data is logically decomposed to smaller pieces in the From-To Table so as to provide more useful information for use in developing virtual cells.

3.6.4 Nondecreasing-From Table

The Nondecreasing-From Table is a table in which the From-To Table is further modified and simplified. In a Nondecreasing-From Table, the machines in the "From" column are arranged in a nondecreasing order and the desired demands of parts using the same machine

Table 3. An example of From-To Table

Parts	From	To	Demand
part 1	0	9	200
part 1	9	7	200
part 1	7	8	200
part 1	8	5	200
part 1	5	4	200
part 1	4	18	200
part 1	18	5	200
part 1	5	6	200
part 1	6	10	200
part 1	10	1	200
part 1	1	2	200
part 1	2	3	200
part 1	3	0	200
part 2	0	11	150
part 2	11	10	150
part 2	10	12	150
part 2	12	7	150
part 2	7	13	150
part 2	13	14	150
part 2	14	15	150
part 2	15	16	150
part 2	16	17	150
part 2	17	1	150
part 2	1	2	150
part 2	2	3	150
part 2	3	0	150
part 3	0	9	325
part 3	9	7	325
part 3	7	8	325
part 3	8	5	325
part 3	5	11	325
part 3	11	10	325
part 3	10	12	325
part 3	12	7	325
part 3	7	13	325
part 3	13	14	325
part 3	14	15	325
part 3	15	16	325
part 3	16	17	325
part 3	17	0	325
part 4	0	9	405
part 4	9	7	405
part 4	7	8	405
part 4	8	5	405
part 4	5	4	405
part 4	4	18	405
part 4	18	5	405
part 4	5	6	405
part 4	6	10	405
part 4	10	13	405
part 4	13	14	405
part 4	14	15	405
part 4	15	16	405
part 4	16	17	405
part 4	17	1	405
part 4	1	2	405
part 4	2	3	405
part 4	3	0	405

pair are summed together in the "Total" column. Furthermore, the dummy machines exhibited in a From-To Table are not shown in a Nondecreasing-From Table. An example of a Nondecreasing-From Table is shown in Table 4, in which the machines in the "From" column are listed in a nondecreasing order. In addition, because parts 1, 3, and 4 use the same machine pair, consisting of flow from machine 9 to machine 7, the desired volume of flow between the two machines is 930 units, which is the sum of the production volumes of the three parts.

3.6.5 Nondecreasing-To Table

Like the Nondecreasing-From Table, the Nondecreasing-To Table is derived from the From-To Table. In a Nondecreasing-To Table, the machines in the "To" column are arranged in a nondecreasing order and the desired demands of parts using the same machine pair are summed together in the "Total" column. Also, the dummy machines are not shown in a Nondecreasing-To Table.

Table 4. A Nondecreasing-From Table

From	To	Total
1	2	755
2	3	755
4	18	605
5	4	605
5	6	605
5	11	325
6	10	605
7	8	930
7	13	475
8	5	930
9	7	930
10	1	200
10	12	475
10	13	405
11	10	475
12	7	475
13	14	880
14	15	880
15	16	880
16	17	880
17	1	555
18	5	605

An example of a Nondecreasing-To Table is shown in Table 5, in which the machines in the "To" column are listed in nondecreasing order. Again, the desired volumes between machine pairs are summed and shown in the "Total" column.

3.6.6 In-degree, Out-degree, and Difference

The terms In-degree and Out-degree, borrowed from graph theory, are redefined here. In this research, the In-degree of machine k is defined as the total number of machines succeeded by machine k in a Nondecreasing-From Table, that is, the count of machines in the "From" column with the machine k in the "To" column in a Nondecreasing-From Table.

The Out-degree of machine k is defined as the total number of machines preceded by machine k in a Nondecreasing-To Table, that is, the count of machines in the "To" column with the machine k in the "From" column in a Nondecreasing-To Table.

The Difference is the difference between the In-degree and Out-degree of a machine and is calculated by the equation: In-degree – Out-degree = Difference.

Table 5. A Nondecreasing-To Table

From	To	Total
10	1	200
17	1	555
1	2	755
2	3	755
5	4	605
8	5	930
18	5	605
5	6	605
9	7	930
12	7	475
7	8	930
6	10	605
11	10	475
5	11	325
10	12	475
7	13	475
10	13	405
13	14	880
14	15	880
15	16	880
16	17	880
4	18	605

For example, the In-degree and Out-degree of the machines in Figure 14 can be counted by using Tables 4 and 5, as shown in Table 6. In the table, the In-degree of machine 1 is equal to 2, since there are two machines (machines 10 and 17) that link to machine 1 in Table 5; the Out-degree of machine 1 is equal to 1, because only one machine (machine 2) is linked from machine 1 in Table 4. Therefore, the Difference for machine 1 is equal to 1 (2–1=1).

3.6.7 Candidate First Machine and Candidate Last Machine

A candidate first machine is a machine that might be located at the first position of a cell. Similarly, a candidate last machine is a machine that might be located at the last position of a cell. To identify candidate first machines and candidate last machines among all machines, the From-To Table is required and the following rules are applied.

Table 6. An example of In-degree and Out-degree

Machine #	In-degree	Out-degree	Difference
1	2	1	1
2	1	1	0
3	1	0	1
4	1	1	0
5	2	3	-1
6	1	1	0
7	2	2	0
8	1	1	0
9	0	1	-1
10	2	3	-1
11	1	1	0
12	1	1	0
13	2	1	1
14	1	1	0
15	1	1	0
16	1	1	0
17	1	1	0
18	1	1	0

Rule 1: If Difference >= 1, and the machine is not succeeded by machine 0, then the machine is a candidate first machine.

Rule 2: If Difference <= -1, and the machine is not preceded by machine 0, then the machine is a candidate last machine.

Rule 3: If Difference >= 1, and the machine is succeeded by machine 0, then the machine is a candidate last machine.

Rule 4: If Difference <= -1, and the machine is preceded by machine 0, then the machine is a candidate first machine.

Rule 5: If Rules 1 and 4 cannot produce any candidate for the first position, then the first machines (without considering the dummy machine) in all part routes will be candidate first machines.

Rule 6: If Rules 2 and 3 cannot produce any candidate for the last position, then the last machines (without considering the dummy machine) in all part routes will be candidate last machines.

For the sake of illustration, machine 1 may be used as an example to demonstrate how to apply these rules. First, the Difference of machine 1 is equal to 1 in Table 6. Next, as shown in Table 3, machine 1 is not succeeded by machine 0. Therefore, Rule 1 is activated, and machine 1 is a candidate first machine. Similarly, machine 9 and machine 13 are also candidate first machines. By using the same technique, one can obtain machines 3, 5, and 10 as candidate last machines.

These candidate first machines and candidate last machines will be used to initiate the creation of a virtual cell. For the sake of convenience, all candidate first machines are arranged in ascending order and given an index, k. For example, there are three candidate first machines, machines 1, 9, and 13, as shown in Table 7. In this case, machine 9 is the second candidate first machine. Therefore, machine 9 has k equal to 2, as shown in Table 7. Moreover, the indicator, K, is employed to indicate the number of candidate first machines. In this case, there are three candidate first machines; therefore, K is equal to 3. Similarly, all

Table 7. An illustration of Candidate first and last machines

k	Candidate first machine	j	Candidate last machine
1	Machine 1	1	Machine 3
2	Machine 9	2	Machine 5
3	Machine 13	3	Machine 10
K=3		J=3	

candidate last machines are arranged in ascending order and each candidate last machine is given an index, j, which indicates the number of candidate last machines.

3.6.8 The Candidate Cell Creation Algorithm

A candidate cell is a cell that has the potential to be a virtual cell itself. The candidate cell creation algorithm is developed to produce candidate cells used in the Ko's virtual cell formation algorithm described in the next section. A candidate cell always has only one candidate first machine and only one candidate last machine among its members. To create a candidate cell, a Nondecreasing-From Table is used to trace the connection between machines. A candidate cell is successfully generated if it starts with one of the candidate first machines and ends with one of the candidate last machines, and if none of its members appears more than once in the cell. The parameters used in candidate cell creation algorithm are given below:

k: the index of candidate first machine

K: the number of candidate first machines

j: the index of candidate last machine

J: the number of candidate last machines

M: the index of counting number

U: the universal machine pool.

C:	the current machine pool.
T:	the temporary machine set.
TL:	the temporary label set.
CC:	the candidate cell.
Set_flag:	the index for a temporary machine set. 1, a candidate cell exists; 0, otherwise.
Number_cell:	the number of candidate cells which have been created
Cell_member:	the number of machines in *CC*.
Current_machine:	the index of machine.

The candidate cell creation algorithm is presented as follows:

Step 1: Use Rules 1-6 to identify all candidate first machines and candidate last machines. Let k and K be the index and the number of candidate first machines, respectively. Let j and J be the index and the number of candidate last machines, respectively.

Step 2: Set k = 1 and Number_cell = 0

Step 3: Set j = 1.

Step 4: 4.a. Store all machines in a universal machine pool, *U*.

4.b. Ignore all candidate first machines and candidate last machines except the k^{th} candidate first machine and the j^{th} candidate last machine in the universal machine pool, *U*.

4.c. Initiate a temporary machine set, *T*. That is, set $T = \phi$.

4.d. Initiate the Set_flag of *T*. That is, set Set_flag = 0.

4.e. Store the k^{th} candidate first machine in the temporary machine set, *T*.

4.f. Deactivate the k^{th} candidate first machine in the universal machine pool, *U*.

4.g. Initiate a temporary label set, *TL*. That is, set $TL = \{0\}$.

4.h. Initiate a current machine pool, *C*. That is, set $C = \phi$.

4.i. Store the k^{th} candidate first machine in the current machine pool, *C*.

Step 5: 5.a. Identify all machines that succeed the machines in the current machine pool, *C*, and are active in the universal machine pool, *U*.

5.b. For each machine X' identified in Step (5.a), place its direct predecessor in the temporary label set, *TL*.

5.c. Place each identified machine X' found in Step (5.a) in the temporary machine set, *T*, with the position of X' in *T* corresponding with the position of its predecessor machine in *TL*.

5.d. Set $C = \phi$.

5.e. Place each identified machine X' found in Step (5.a) in the current machine pool, *C*.

5.f. Deactivate each identified machine X' in the universal machine pool, *U*.

Step 6: Evaluate each machine in the current machine pool, *C*, as follows:

6.a. If the machine is the j^{th} candidate last machine,
then set the Set_flag = 1 and go to Step 7.

6.b. If there is no machine in the current machine pool, *C*,
then set the Set_flag = 0 and go to Step 8.

6.c. If Step (6.a) and Step (6.b) are not applied, then go to Step 5.

Step 7: 7.a. Obtain the candidate cell by using *T* and *TL* at hand as follows:

7.a.1 Initiate a candidate cell, *CC*. That is, set $CC = \phi$.

7.a.2 Set Cell_member = 0.

7.a.3 Extract the j^{th} candidate last machine from
the temporary machine set, *T*.

7.a.4 Set the j^{th} candidate last machine as the Current_machine

7.a.5 Store the Current_machine in the candidate cell, *CC*, and
Set Cell_member = Cell_member + 1.

7.a.6 Identify the machine Y' that directly precedes the Current_machine
in the temporary label set, *TL*.

7.a.7 If the machine Y' identified in Step (7.a.6) is machine 0, then output the candidate cell, *CC*, with the associated Cell_member and go to Step (7.b).

7.a.8 Extract the machine Y' identified in Step (7.a.6) from the temporary machine set, *T*. Set the machine Y' as the Current_machine. Go to Step (7.a.5).

7.b. Set Number_cell = Number_cell + 1.

Step 8: If j = J, then go to Step 9. Otherwise, set j = j + 1 and go to Step 4.

Step 9: If k = K, then go to Step 10. Otherwise, set k = k + 1 and go to Step 3.

Step 10: Terminate. The number of candidate cells created is equal to Number_cell.

For the sake of illustration, machine 1 and machine 3, which in Table 7 are the first candidate first machine and the first candidate last machine, respectively, are employed to demonstrate how to operate the candidate cell creation procedure. With machine 1 and machine 3 on hand, the procedure starts from Step 3. The detailed illustration is given in Appendix A. The generated candidate cell is **[m1, m2, m3]**.

In fact, if there are K candidate first machines and J candidate last machines, the creation procedure will repeat K * J times. However, it is possible that not all the K * J combinations can produce a candidate cell.

Intersection Rule

The concept of the intersection rule is borrowed from set theory, and the situation is described as follows: Suppose several candidate cells have the same two machines at their first two positions or at their last two positions. Then the two machines are the intersection of these candidate cells. Therefore, the intersection rule is applied to extract the two machines. Then, a new candidate cell is created with the two machines, and the original candidate cells are eliminated.

For example, suppose there are three candidate cells, [m4, m7, m8, m9, m3], [m4, m7, m8, m10, m5], and [m4, m7, m12, m11, m13] on hand. Obviously, the three candidate cells have the same machines, machine 4 and machine 7, at their first two

positions. Therefore, the intersection rule is applied and a new candidate cell, [**m4, m7**], is created. Then, the original three candidate cells are eliminated.

Subset Rule

The concept of the subset rule is borrowed from set theory, and the situation can be described as follows: If two candidate cells exist, and one is a subset of the other, then the larger candidate cell eliminates the smaller one.

For example, suppose there are two candidate cells, [m11, m13] and [m11, m12, m13]. If the subset rule is applied, the smaller candidate cell, [m11, m13], is eliminated by the larger candidate cell, [**m11, m12, m13**].

Union Rule

The concept of the union rule is also borrowed from set theory. Three situations are considered in this rule. The first is for those candidate cells that are the same size and whose number of machines is greater than or equal to 3. If two such candidate cells are different from each other in only one machine, then the two candidate cells could be combined to form a new candidate cell. The new candidate cell will therefore replace the two original candidate cells. For instance, suppose there are two candidate cells, [m3, m2, m5] and [m4, m2, m5]; then the new candidate cell [**m3, m4, m2, m5**] is created to replace the two original candidate cells.

The second situation is for the two-machine candidate cells. Suppose two such candidate cells share one machine but differ with regard to the other machine. In addition, there is a candidate cell that consists of the two other machines only. Then, the three two-machine candidate cells are merged to form a new candidate cell, which will replace the three original candidate cells. For instance, suppose there are two candidate cells, [m2 m5] and [m6 m5], and another candidate cell, [m2 m6], exists. Then, the three candidate cells are merged to form a new candidate cell, [m2 m6 m5], which replaces the three original two-machine candidate cells.

The third situation considered in the rule is with regard to the job routings represented by candidate cells. If a pair of candidate cells is observed to have the same sequence pattern in

the job routings, the two candidate cells could be married together. In other words, suppose candidate cell A' always precedes candidate cell B' and B' always succeeds A'. Then, the two candidate cells will be married together to form a new candidate cell, which consists of all machines of its parents and replaces its parents in the job routings. For instance, suppose it is observed that cell 5 always precedes cell 2 and cell 2 always succeeds cell 5 in the job routings. Then, a new candidate cell consisting of all machines in cell 5 and cell 2 is created to replace cell 5 and cell 2 in the job routings.

Split Rule

The split rule is employed to decompose a candidate cell into several two-machine candidate cells. The rule is activated when no block of consecutive machines in the job routings can fully match any of the newly created candidate cells. Under this condition, the job routings could not be updated; as a result, no more candidate cells would be produced. The problem might be solved by activating the split rule; that is, the candidate cells are further decomposed into several two-machine candidate cells. In this way, it is guaranteed that at least one of the two-machine candidate cells could update the job routings.

For instance, suppose the candidate cell creation algorithm generates only a candidate cell, [m1, m4, m6] that no block of consecutive machines in the job routings can fully match. If the split rule is activated and the candidate cell is further separated into 2 two-machine candidate cells, [m1, m4] and [m4, m6], at least one of the two-machine candidate cells can update the job routings.

3.7 The Ko's virtual cell Formation Algorithm

The Ko's virtual cell formation algorithm could be represented by a flow chart as shown in Figure 17. The proposed algorithm is described below.

Step 1: Input the required data (that is, a set of job routings with the desired demands for the jobs).

Step 2: Create the From-To Table, Nondecreasing-From Table, and Nondecreasing-To Table by using the current job routings.

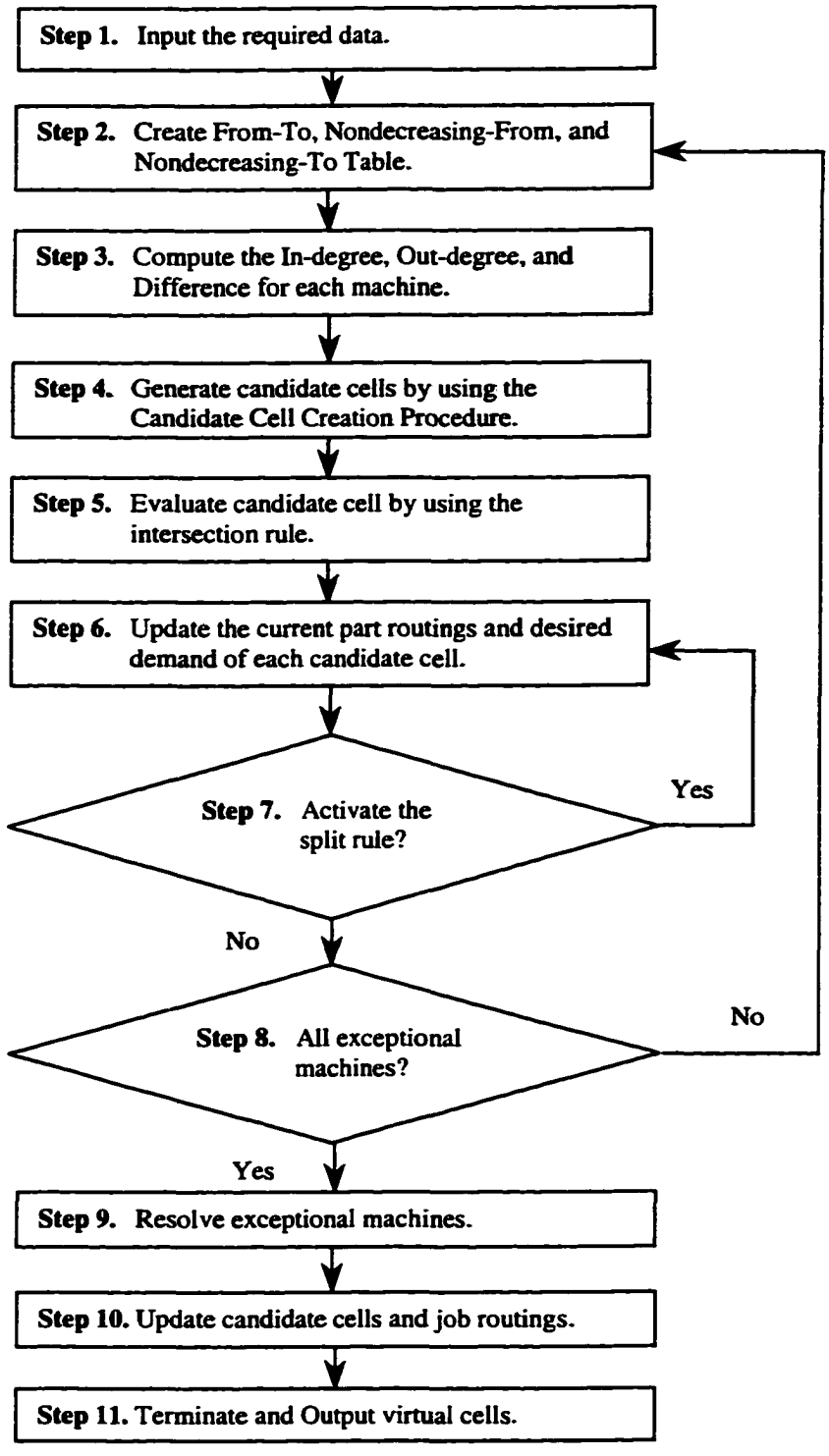


Figure 17. Flow chart of the Ko's virtual cell formation procedure

Step 3: Compute the In-degree, Out-degree, and Difference for each machine, except the dummy machine (machine 0).

Step 4: Generate candidate cells by invoking the Candidate Cell Creation algorithm.

Step 5: Evaluate candidate cells by using the intersection rule.

Step 6: Update the current job routings and the desired demand of candidate cells as follows:

6.a. Let K be the number of newly created candidate cells and let J be the number of jobs.

6.b. Arrange the candidate cells on hand in a nonincreasing order based on their sizes. Assign each candidate cell an index, k, according to the order. Assign an index, j, for each job.

6.c. Set m = 0.

6.d. Set k = 1.

6.e. Set j = 1.

6.f. If there is a block of consecutive machines on the routing of job j that is fully contained in candidate cell k or matches candidate cell k:

6.f.1. Replace the matching block of machines by a dummy machine.

6.f.2. Update the volume of production for candidate cell k by adding the volume of production for job j to the volume of production for candidate cell k.

6.f.3. Set m = m + 1

6.g. If two or more dummy machines consecutively appear in a job routing, then reduce these consecutive dummy machines to one dummy machine only.

6.h. If j = J, go to Step (6.i); otherwise, set j = j + 1 and go to Step (6.f).

6.i. If k = K, then go to Step 7; otherwise, set k = k + 1 and go to Step (6.e).

Step 7 7.a. If m is equal to 0, then:

7.a.1 Activate the split rule to produce two-machine candidate cells.

7.a.2 Set K equal to the number of candidate cells created in Step (7.a.1).

7.a.3 Go to Step 6.

7.b. Otherwise, go to Step 8.

Step 8: Evaluate the updated job routings as follows:

8.a. If the machines existing in the updated job routings are all exceptional machines, then go to Step 9.

8.b. Otherwise, go to Step 2 with the updated job routings.

Step 9: Resolve the exceptional machines as follows:

9.a. Evaluate all candidate cells for each exceptional machine Z'.

9.a.1 Identify all the candidate cells that contain the exceptional machine Z'.

9.b.2 Of the candidate cells identified in Step (9.a.1), replace the exceptional machine Z' by the candidate cell with the least total production volume.

9.b. If Step (9.a) is not applied, then follow the procedure below:

9.b.1 Identify candidate cells that the exceptional machine Z' either directly precedes or directly succeeds on the job routings.

9.b.2 Choose the candidate cell C' with the least production volume from the identified candidate cells in Step (9.b.1).

9.b.3 Expand the candidate cell C' to include the exceptional machine Z' as its first machine if the exceptional machine Z' directly precedes C'. Otherwise, expand the candidate cell C' to include the exceptional machine Z' as its last machine if the exceptional machine Z' directly succeeds C'.

Step 10: 10.a. Update all candidate cells by using the subset rule and union rule.

10.b. Update the job routings that are represented by candidate cells by using the union rule.

10.c. Update the production volumes of candidate cells.

Step 11: Terminate. The candidate cells on hand are the virtual cells and the job routings have already been represented by candidate cells.

Through use of the Ko's virtual cell formation procedure, five virtual cells are generated for the example in Figure 14. The five virtual cells are shown in Figure 18. Note that no exceptional position machine is produced during the process. As shown in Figure 18, the sharing concept has been well demonstrated in the cell formation procedure. For example, cell 2 and cell 4 share machine 5, and virtual cell 2 serves parts 1, 3, and 4. The detailed cell formation procedure for this example is presented in Appendix B. In addition, the shop layout with these virtual cells is conceptually represented in Figure 19. As shown in the figure, some machines belonging to the same cell are close to each other while others are not. For example, machines 1, 2, and 3 in virtual cell 1 are neighbors in the shop, but machine 7 is far from the other machines in virtual cell 5.

3.8 Test Results

The Ko's virtual cell formation algorithm has been applied to some test problems that have appeared in the literature. The first example is shown in Figure 20 [186]. There are five jobs to be produced, and sixteen machines in the shop. After Step 8, there were two exceptional machines (machine 7 in part 2 and machine 1 in part 4), as shown in Figure 21.

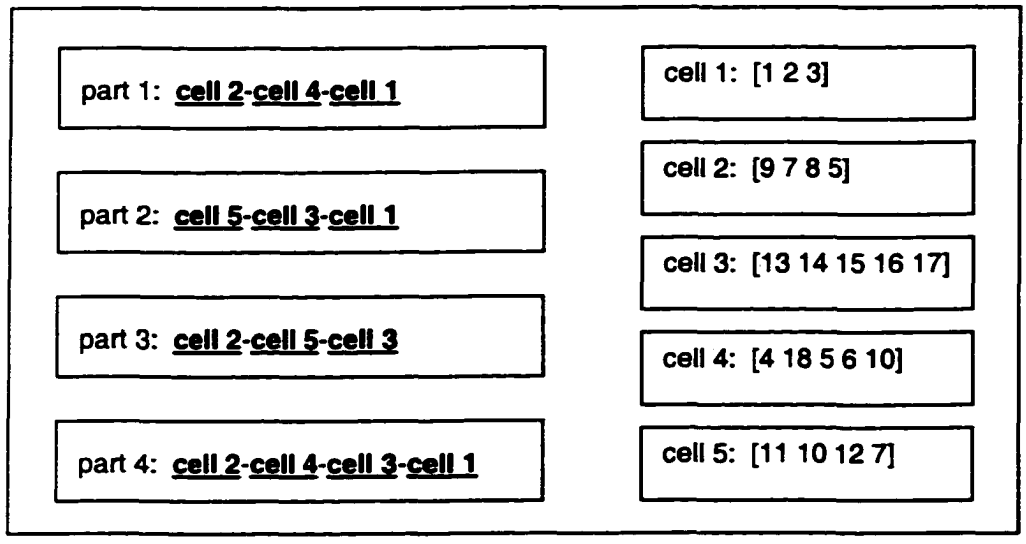


Figure 18. The generated virtual cells

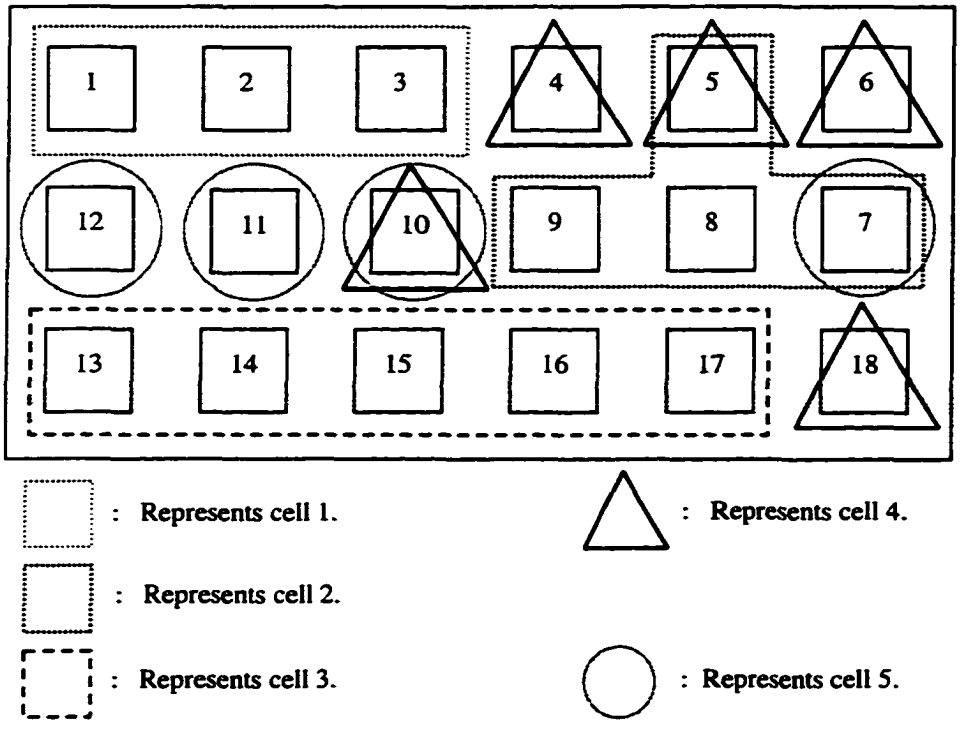


Figure 19. The shop layout with 5 virtual cells

Part 1: 1-6-16-11-5-7	Desired demand: 34
Part 2: 7-12-8-3-4-15	Desired demand: 20
Part 3: 15-14-3-12-8	Desired demand: 45
Part 4: 1-2-11-6-14-4-10	Desired demand: 25
Part 5: 1-6-2-9-13-15	Desired demand: 15

Figure 20. The machine routes and desired demands of the first example

Part 1: cell 1-cell 5-cell 6	Cell 1: [1 6]	Cell 6: [5 7]
Part 2: 7-cell 2-cell 8	Cell 2: [12 8]	Cell 7: [6 14 4 10]
Part 3: cell 9-cell 2	Cell 3: [2 9 13 15]	Cell 8: [3 4 15]
Part 4: 1-cell 4-cell 7	Cell 4: [2 11]	Cell 9: [15 14 3]
Part 5: cell 1-cell 3	Cell 5: [16 11]	

Figure 21. The intermediate result of the first example with exceptional machines

Because machines 7 and 1 appear in cells 6 and 1, respectively, the two cells now replace the two exceptional machines in the job routings, as shown in Figure 22. The figure shows that cell 4 precedes cell 7 only and cell 7 succeeds cell 4 only, so the two candidate cells are married together to form a new candidate cell. The candidate cells and the job routings are then updated again, as shown in Figure 23.

The second example [97] is shown in Figure 24. There are 16 parts and 12 machines. The intermediate result with exceptional machines is shown in Figure 25; the exceptional machines are machines 1, 6, 7, 9, 10, and 12. Based on Step (8.a), each exceptional machine in the job routings is replaced by an associated candidate cell, as shown in Figure 26. Moreover, the candidate cells and job routings in the figure can be further updated by using the subset rule and union rule. The final result of this example is as shown in Figure 27. In the figure, there are nine cells, and these nine virtual cells represent the job routings.

3.9 Discussion

In this chapter, a virtual cell formation procedure was presented. The sharing concepts in virtual cellular manufacturing were considered in the algorithm development. The performance of the proposed procedure was demonstrated. With the virtual cell formation

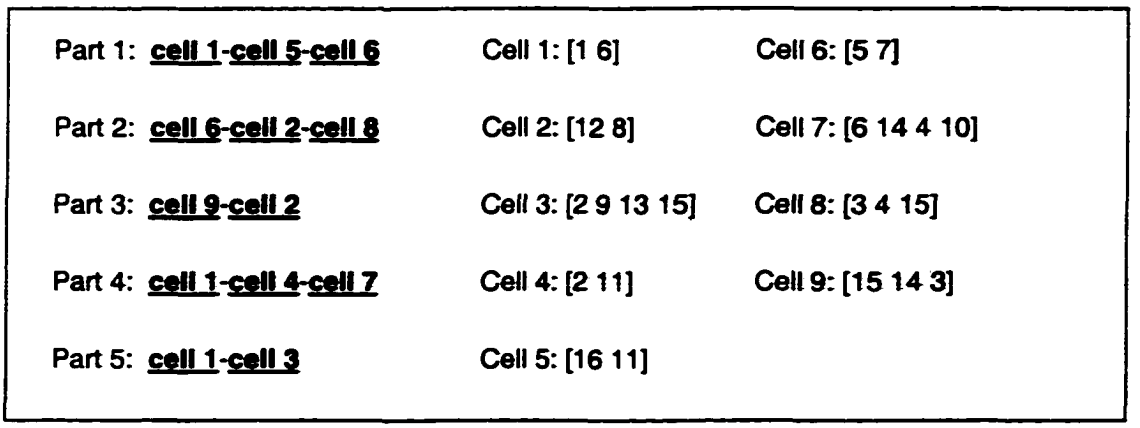


Figure 22. The intermediate result of the first example without exceptional machines

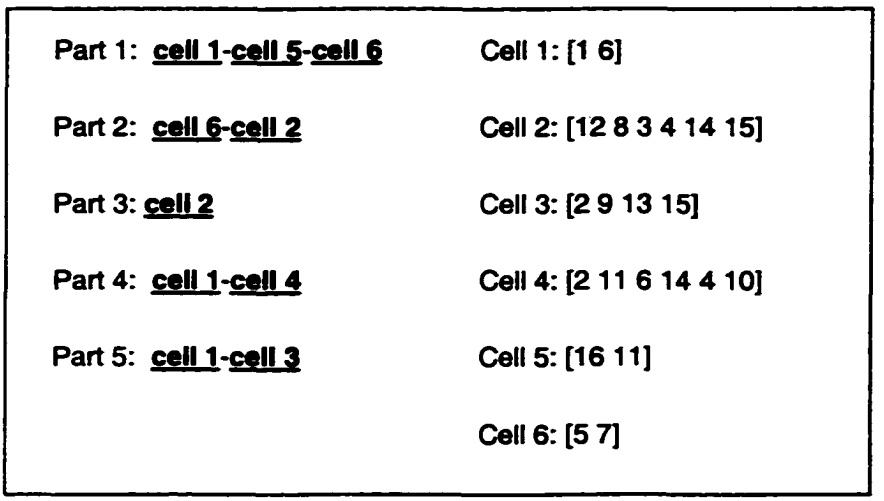


Figure 23. The final result of the first example

Part 1: 1-4-8-9	Desired demand: 200
Part 2: 1-4-7-4-8-7	Desired demand: 300
Part 3: 1-2-4-7-8-9	Desired demand: 100
Part 4: 1-4-7-9	Desired demand: 300
Part 5: 1-6-10-7-9	Desired demand: 200
Part 6: 6-10-7-8-9	Desired demand: 100
Part 7: 6-4-8-9	Desired demand: 200
Part 8: 3-5-2-6-4-8-9	Desired demand: 100
Part 9: 3-5-6-4-8-9	Desired demand: 100
Part 10: 4-7-4-8	Desired demand: 200
Part 11: 11-7-12	Desired demand: 100
Part 12: 11-12	Desired demand: 100
Part 13: 11-7-10	Desired demand: 300
Part 14: 1-7-11-10-11-12	Desired demand: 300
Part 15: 11-7-12	Desired demand: 100
Part 16: 6-7-10	Desired demand: 300

Figure 24. The machine routings and desired demands of the second example

Part 1: 1-cell 3	Cell 1: [1 6 10]
Part 2: 1-cell 7-cell 9-7	Cell 2: [3 5]
Part 3: cell 15-cell 7-cell 14	Cell 3: [4 8 9]
Part 4: 1-cell 7-9	Cell 4: [11 10]
Part 5: cell 1-cell 12	Cell 5: [11 12]
Part 6: cell 8-cell 13	Cell 6: [1 7]
Part 7: 6-cell 3	Cell 7: [4 7]
Part 8: cell 2-cell 16-cell 3	Cell 8: [6 10]
Part 9: cell 2-6-cell 3	Cell 9: [4 8]
Part 10: cell 7-cell 9	Cell 10: [6 7]
Part 11: cell 11-12	Cell 11: [11 7]
Part 12: cell 5	Cell 12: [7 9]
Part 13: cell 11-10	Cell 13: [7 8 9]
Part 14: cell 6-cell 4-cell 5	Cell 14: [8 9]
Part 15: cell 11-12	Cell 15: [1 2]
Part 16: cell 10-10	Cell 16: [2 6]

Figure 25. The intermediate result of the second example with exceptional machines

Part 1: cell 15-cell 3	Cell 1: [1 6 10]
Part 2: cell 1-cell 7-cell 9-cell 13	Cell 2: [3 5]
Part 3: cell 15-cell 7-cell 14	Cell 3: [4 8 9]
Part 4: cell 6-cell 7-cell 14	Cell 4: [11 10]
Part 5: cell 1-cell 12	Cell 5: [11 12]
Part 6: cell 8-cell 13	Cell 6: [1 7]
Part 7: cell 8-cell 3	Cell 7: [4 7]
Part 8: cell 2-cell 16-cell 3	Cell 8: [6 10]
Part 9: cell 2-cell 16-cell 3	Cell 9: [4 8]
Part 10: cell 7-cell 9	Cell 10: [6 7]
Part 11: cell 11-cell 5	Cell 11: [11 7]
Part 12: cell 5	Cell 12: [7 9]
Part 13: cell 11-cell 4	Cell 13: [7 8 9]
Part 14: cell 6-cell 4-cell 5	Cell 14: [8 9]
Part 15: cell 11-cell 5	Cell 15: [1 2]
Part 16: cell 10-cell 8	Cell 16: [2 6]

Figure 26. The intermediate result of the second example without exceptional machines

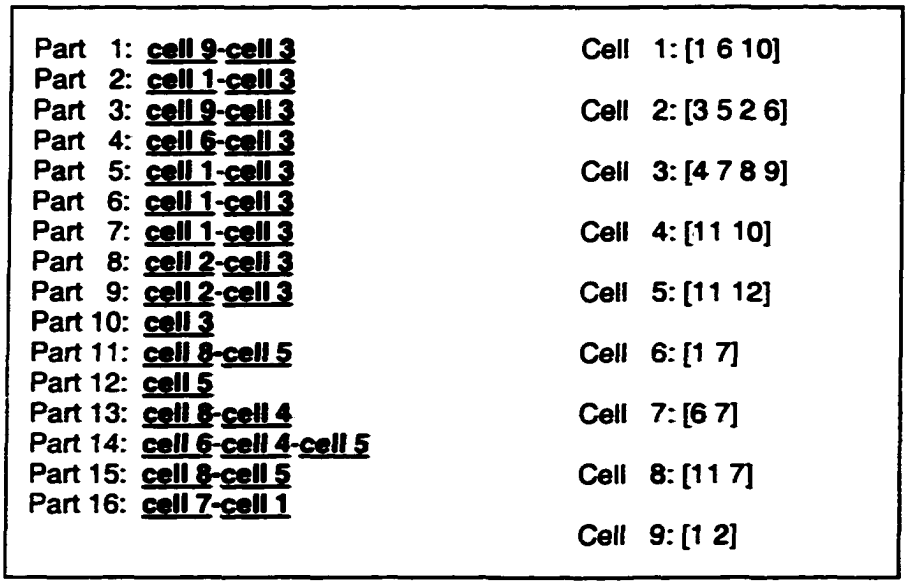


Figure 27. The final result of the second example

procedure, virtual cellular configuration is ready to be used in the processing system configuration module.

The other module in a virtual production system, the networking module, helps minimize the material handling distance traveled in a production session. Under the assumption that machines are not movable in a shop, the machine connections would be performed by a material handling system. In this study, automated guided vehicles (AGVs) are chosen as the preferred mode of material handling. AGVs not only connect machines in a shop, but also play a role in linking machines belonging to the same cell. The following chapter will discuss the AGV guidepath network design that has the objective of minimizing material handling distance.

CHAPTER 4. AGV GUIDEPATH NETWORK DESIGN

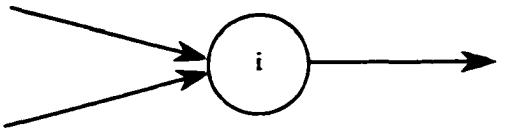
The second required module in a virtual production system is the networking module. Because the frozen position of machines is assumed, the networking module performs the task of linking the machines by the material handling system through an efficient network design. The material handling system employed in the networking module is an AGVS (Automated Guided Vehicle System) within a virtual AGV guidepath network. It is called "virtual" because there are no physical taps or wires on the ground and the flow network exists as a database type in a computer. In practice, AGVs are guided by using radio or laser beams, and follow an associated guidepath network to link machines together. In a virtual AGV guidepath network, the direction of traffic flow on an aisle segment is not fixed, but can be changed from one production session to another. In any given production session, the traffic flow directions are set to respond to changes in product mix and routing pattern.

In this chapter, an AGV guidepath network design procedure is proposed. The objective of the procedure is to design a network that minimizes the total material handling cost in terms of distance traveled for a production instance. Under a dynamic production environment, the AGV guidepath network needs to be updated in response to product mix changes over time.

4.1 Introduction

A network consists of two components, nodes and arcs. Two nodes are neighbors if and only if an arc in the network connects them. An arc is considered to be an input arc if it points to and terminates at the node. In a similar manner, an output arc starts at and points away from the node. An example of a node with its input arcs and output arcs is as shown in Figure 28. In the figure, node i has two input arcs and one output arc.

In a feasible network, each node must have at least one input arc and one output arc. An example of a feasible network is as shown in Figure 29. In addition, a path (a chain of several arcs that connects two nodes in the network) can be seen in Figure 29; the path that connects node 1 and node 3 consists of two arcs, Arc $_{1,2}$ and Arc $_{2,3}$. In this research, AGVs



Two input arcs to node i One output arc from node i

Figure 28. An example of input and output arcs

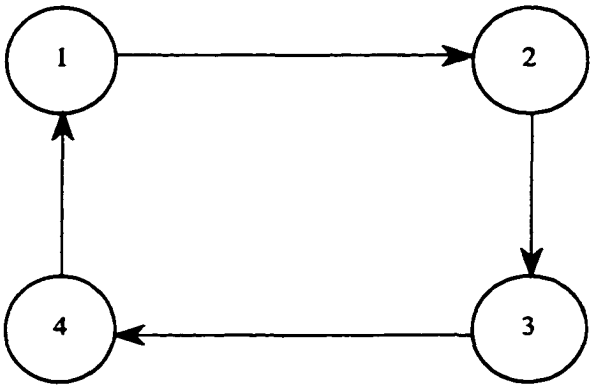


Figure 29. An example of a feasible network

are employed as the material handling system, while the network is used to guide the movement of the vehicles. This is how the name "AGV guidepath" originated.

A shop floor can be represented as a set of nodes and undirected arcs (edges), as shown in Figure 30 [27]. A continuous space is partitioned into non-overlapping blocks that represent machines or workstations. Edges represent the boundaries between blocks, and nodes represent pick-up and drop-off points of workstations (machines) and intersections between edges. The pick-up and drop-off points of a workstation are the load and unload points of materials for the workstation. An example of a directed AGV guidepath network of a shop is represented in Figure 31, in which there are 4 workstations (machines), 14 nodes, and 17 arcs.

An AGV travels through a guidepath to connect a pair of points or locations in a shop. For example, in Figure 31, one possible path linking the pick-up point of machine 1 and the drop-off point of machine 2 involves six arcs and is represented as: Arc $_{1,0} \rightarrow$ Arc $_{0,2} \rightarrow$ Arc $_{2,10} \rightarrow$ Arc $_{10,3} \rightarrow$ Arc $_{3,12} \rightarrow$ Arc $_{12,4}$. The guidepath starts from the pick-up point of

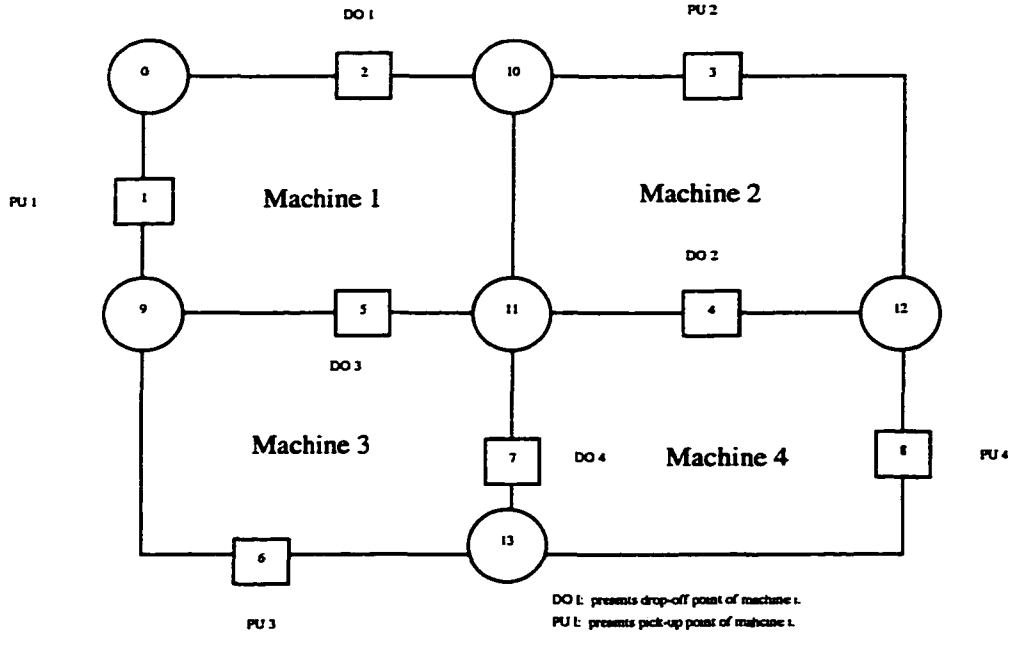


Figure 30. A shop represented as a set of nodes and undirected arcs

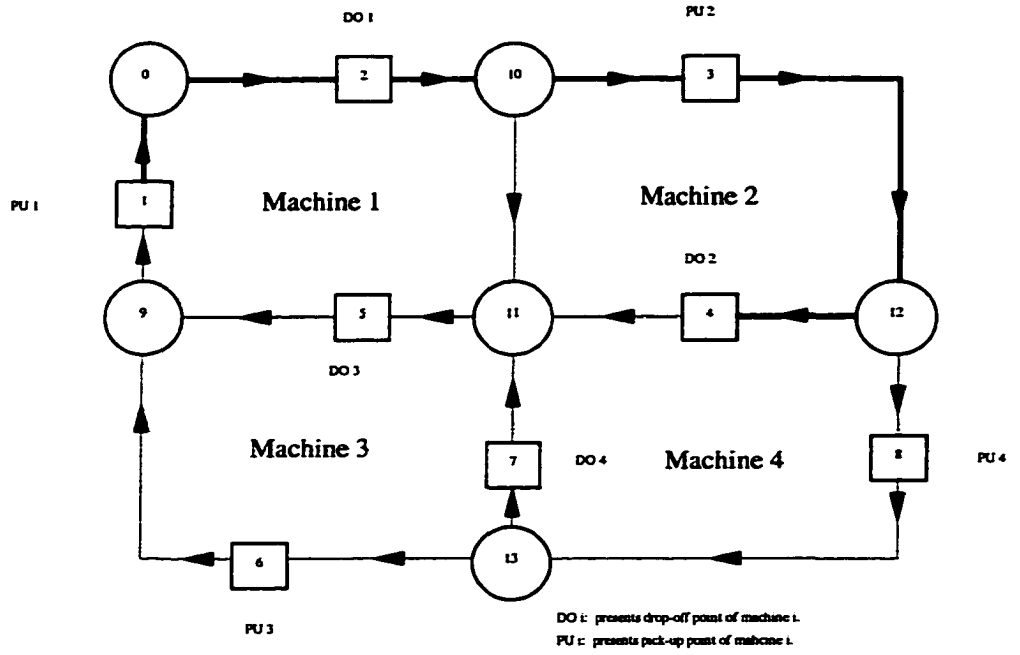


Figure 31. An AGV guidepath network

machine 1, that is, node 1, passes five intermediate nodes (nodes 0, 2, 10, 3, and 12), and terminates at the drop-off point of machine 2, that is, node 4.

The chapter is organized as follows. The required input data, terminology, and techniques are described and prepared in Sections 4.2 and 4.3. Then, the proposed AGV guidepath design procedure is presented in Section 4.4. Finally, the initial test results and discussion are presented and discussed in Sections 4.5 and 4.6, respectively.

4.2 Input Data

To design an AGV guidepath network, two types of input data are needed: a distance matrix file and a flow volume file. The distance matrix file represents a shop layout and provides information about the distances between nodes. In a distance matrix file, if two nodes i and j are directly connected by an edge, then the matrix entry (i, j) is filled with the distance between the two nodes. Otherwise, the symbol "∞" is entered. Here, "∞" implies that no direct connection exists between the two nodes.

For example, Table 8 might be the distance matrix file of the shop in Figure 30. Because node 2 and node 10 are directly connected by an edge, the distance measure, 1 unit, is entered as shown in Table 8.

The other needed input data, a flow volume file, provides two kinds of information. One is the flow between machines, while the other is the priority of all flows. Usually, the flow with the largest volume is listed first and has the highest priority in a flow volume file. For example, as shown in Table 9, the flow volume between machines 2 and 4 is 545 units and has the 5th priority in the file. The volume of flow represents the number of material handling moves between two machines. If the volume of flow is not given directly in number of moves, it must be converted to moves before it is entered in the matrix.

4.3 Terminology and Techniques

The terminology and techniques employed in the proposed procedure are defined in this section.

Table 8. An example of a distance matrix file

Nodes	0	1	**2**	3	4	5	6	7	8	9	**10**	11	12	13
0	-	1	2	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
1	1	-	∞	∞	∞	∞	∞	∞	∞	1	∞	∞	∞	∞
2	2	∞	-	∞	∞	∞	∞	∞	∞	∞	**1**	∞	∞	∞
3	∞	∞	∞	-	∞	∞	∞	∞	∞	∞	2	∞	4	∞
4	∞	∞	∞	∞	-	∞	∞	∞	∞	∞	∞	2	2	∞
5	∞	∞	∞	∞	∞	-	∞	∞	∞	2	∞	1	∞	∞
6	∞	∞	∞	∞	∞	∞	-	∞	∞	4	∞	∞	∞	2
7	∞	∞	∞	∞	∞	∞	∞	-	∞	∞	∞	2	∞	1
8	∞	∞	∞	∞	∞	∞	∞	∞	-	∞	∞	∞	2	5
9	∞	∞	∞	∞	∞	2	4	∞	∞	-	∞	∞	∞	∞
10	∞	∞	**1**	2	∞	∞	∞	∞	∞	∞	-	2	∞	∞
11	∞	∞	∞	∞	2	1	∞	2	∞	∞	2	-	∞	∞
12	∞	∞	∞	∞	2	∞	∞	∞	2	∞	∞	∞	-	∞
13	∞	∞	∞	∞	∞	∞	2	1	5	∞	∞	∞	∞	-

Table 9. The flow volumes between workstations

Flow #	From Machine #	To Machine #	Volume	Priority
1	2	1	835	1
2	3	2	780	2
3	1	3	777	3
4	3	4	558	4
5	2	4	545	5
6	4	1	389	6

4.3.1 Terminology

v_i: the node i.

S: the set of arcs with fixed directions.

Arc $_{i,j}$: the directed arc that starts at node i and ends in node j.

D $_{i,j}$: the distance measure of the Arc $_{i,j}$.

FD $_{i,j}$: the flow distance of Arc $_{i,j}$ for each flow.
FD $_{i,j}$ = the flow volume from v_i to v_j * D $_{i,j}$.

AFD $_{i,j}$: the accumulated flow distance of Arc $_{i,j}$ for all flows.
AFD $_{i,j}$ = AFD $_{i,j}$ + FD $_{i,j}$

E(i, j): the difference between AFD $_{i,j}$ and AFD $_{j,i}$, that is,
E(i, j) = AFD $_{i,j}$ - AFD $_{j,i}$

Total_Flow_Distance: the total flow distance based on a network and the associated flow volume file.

In-degree of node i: the number of input arcs into a node i.

Out-degree of node i: the number of output arcs from a node i.

Total-degree of node i: the sum of the In-degree and Out-degree of node i.

Heavy node: a node having a Total-degree of 3 or more.

Heavy arc: an arc connecting two heavy nodes.

For example, in Figure 31, the arc that leads from node 2 to node 10 is represented as Arc $_{2,10}$ and D $_{2,10}$ is equal to 1 unit, as shown in Table 8. The In-degree of node 10 is equal to 1, because of the input arc, Arc $_{2,10}$. The Out-degree of node 10 is equal to 2, because of the two output arcs, Arc $_{10,3}$ and Arc $_{10,11}$. Because the total-degree of node 10 is equal to 3, node 10 is a heavy node. Because node 11 is also a heavy node, Arc $_{11,10}$, is a heavy arc.

4.3.2 Dijkstra's Algorithm

Dijkstra's algorithm [187] is employed in the research to find the shortest path between a pair of machines or nodes. Given a distance matrix for a graph G(V, A) (a graph is defined in Chapter 3), the shortest path from a node v_p to another node v_q is obtained by using this algorithm. Here, A represents the set of directed arcs in a graph.

The set of vertices in V is partitioned into two subsets T and W. The subset T is a vector containing vertices that have permanent labels associated with them. A permanent label of a vertex is the minimum distance of the vertex from v_p, the starting node. The distance is stored in the vector PL. Thus, associated with T is a distance vector PL. The set W is simply $(V - T)$. The vector P contains the vertices adjacent to the nodes in T along the minimum path. The set TL defines the temporary labels associated with vertices in W.

Dijkstra's Algorithm is presented as follows [187]:

Step 1. Initially $T = \{v_p\}$, $PL = \{0\}$, and $P=\{0\}$. A temporary label of ∞ is associated with all vertices in W.

Step 2. Using the distance matrix, determine the vertices in W that are adjacent to any vertex in T. Assign each of these vertices a temporary label equal to the distance of that vertex from v_p. The distance of vertex v_i in W adjacent to v_j in T is its new temporary label given by:

$$\text{Temporary label of } v_i = \min_j [\text{permanent label of } v_j + D_{j,i}]$$

Step 3. Make the smallest temporary label permanent. Transfer the corresponding vertex v_i from W to T. Include in P the vertex v_j (in T) that is adjacent to it. Reset all temporary labels to ∞. Repeat Step 2 and 3 until v_q is included in T or there are no vertices in W that are adjacent to those in T.

Step 4. If v_q is included in *T*, go to Step 5; otherwise, the desired path does not exist; stop.

Step 5. The permanent label of v_q gives the distance of the shortest path from v_p to v_q.

To trace the shortest path, start from v_q in *T*, and identify the corresponding adjacent vertex v_m in *P*. Next, check v_m in *T*, and read out the corresponding preceding adjacent vertex v_n in *P*. Repeat this process until v_p is reached. The traced sequence is the path from v_p to v_q.

The network shown in Figure 31 and the distance matrix in Table 8 are used to demonstrate the operation of Dijkstra's Algorithm. The task is to find the shortest path from the pick-up point of machine 1 to the drop-off point of machine 3; that is, the shortest path from node 1 to node 5. The detailed procedure for applying Dijkstra's Algorithm is presented in Appendix C. According to the solution, the shortest path passes through nodes 1, 0, 2, 10, 11, and 5. The shortest path can be identified as: Arc $_{1,0}$ → Arc $_{0,2}$ → Arc $_{2,10}$ → Arc $_{10,11}$ → Arc $_{11,5}$, represented as boldface lines in Figure 32. In addition, the shortest

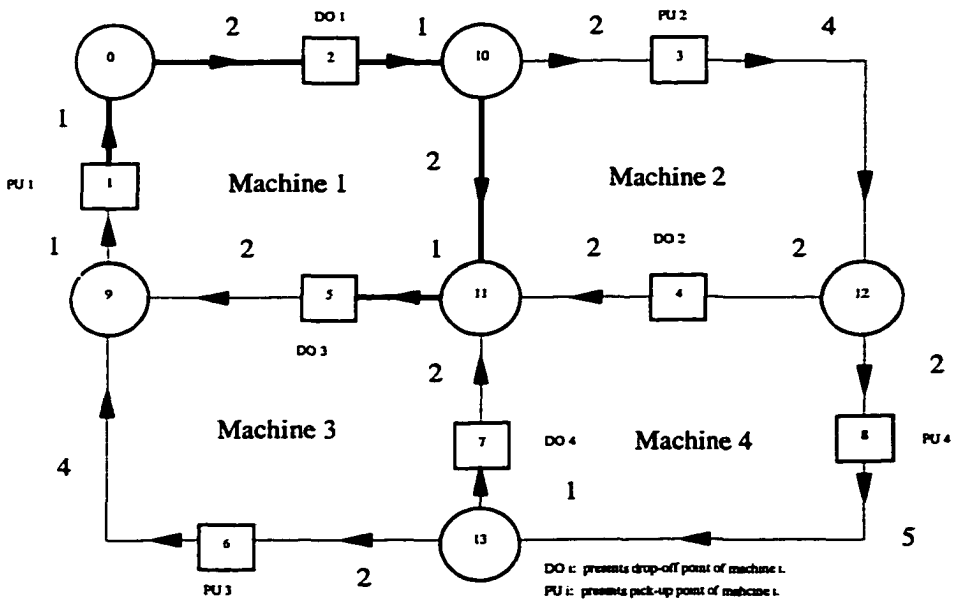


Figure 32. The operation of Dijkstra's Algorithm

distance between nodes 1 and 5 is the sum of $D_{1,0}$, $D_{0,2}$, $D_{2,10}$, $D_{10,11}$, and $D_{11,5}$, which is 7 units.

4.3.3 The Pre-process Algorithm

The Pre-process algorithm is employed to evaluate all segments of an undirected network by use of the flow volume file before the execution of the Complete algorithm, which will be presented later in this section. An undirected network is a network in which no segment is assigned a direction, as shown in Figure 30. The purpose of the Pre-process algorithm is to fix a segment in the direction that ensures that the total flow volume of the flows with lower priority exceeds the flow volume of a flow with higher priority. Thereafter, a preprocessed network is ready for further use in the Complete algorithm. Ultimately, the objective is to convert the undirected network to a directed network. The input data for the Pre-process algorithm are the distance matrix file and the flow volume file.

The Pre-process algorithm is presented as follows:

Step 1: Let K be the number of flows in the flow volume file and k be the priority index of flows.

Step 2: Initiate S, a set of arcs with fixed directions. That is, set $S = \phi$.

Step 3: Fix the directions of all arcs in S on the current network.

Step 4: Set $FD_{i,j} = 0$ and $AFD_{i,j} = 0$ for all edges,
except the edges represented by arcs in S.

Step 5: Set k=1

Step 6: Find the shortest path for the flow with priority k on the current network.

6.a. Use Dijkstra's Algorithm to find the shortest path for the flow with priority k.

6.b. If the shortest path has at least one heavy arc, then try to find alternative paths with the same shortest distance by inactivating one of the heavy arcs in the path already found.

6.b.1 If the alternatives have fewer heavy arcs, then choose the alternative with the fewest heavy arcs as the shortest path for the flow.

6.b.2 If no alternative exists or Step (6.b.1) is not applied, then use the original shortest path as the shortest path for the flow.

6.c. For each arc on the shortest path that has not been used by an earlier flow, assign the priority k to the arc.

6.d. Calculate $FD_{i,j}$ for all arcs in the shortest path of flow k based on the flow volume of k, where $FD_{i,j}$ = flow volume of k * D_{ij}.

6.e. Accumulate $AFD_{i,j}$ for all arcs in the shortest path, where $AFD_{i,j} = AFD_{i,j} + FD_{i,j}$

Step 7: If k = K, then go to Step 8. Otherwise, set k = k + 1 and go to Step 6.

Step 8: Evaluate E(i, j) for all edges (except the edges represented by arcs in *S*). That is,

$$E(i, j) = AFD_{i,j} - AFD_{j,i}, \forall e_{ij} \text{ and } e_{ji}$$

If $Arc_{i,j}$ has higher priority than $Arc_{j,i}$, then E(i, j) should be greater than or equal to 0. Otherwise, a violation exists.

8.a. If no violation exists, then go to Step 9.

8.b. If Step (8.a) is not applied, then

8.b.1 Initiate a set, *V*. That is, set $V = \phi$.

8.b.2 Place all the arc pairs with violation in the set, *V*.

8.b.3 Choose the pair of arcs with the largest difference between the accumulated flow distances in *V*. That is,

$$E(m, n) = \max\{|E(i, j)|\}, \qquad \forall E(i, j) \text{ in } V$$

8.b.4 If $AFD_{m,n} > AFD_{n,m}$, then store $Arc_{m,n}$ in *S*.

8.b.5 Otherwise, store $Arc_{n,m}$ in *S*.

8.b.6 Go to Step 3

Step 9: Output the pre-process network. The output of this algorithm is used by the Complete algorithm.

An example using the distance matrix in Table 8 and the flow volume file in Table 9 might have the pre-processed network shown in Figure 33. Note here, the arcs including

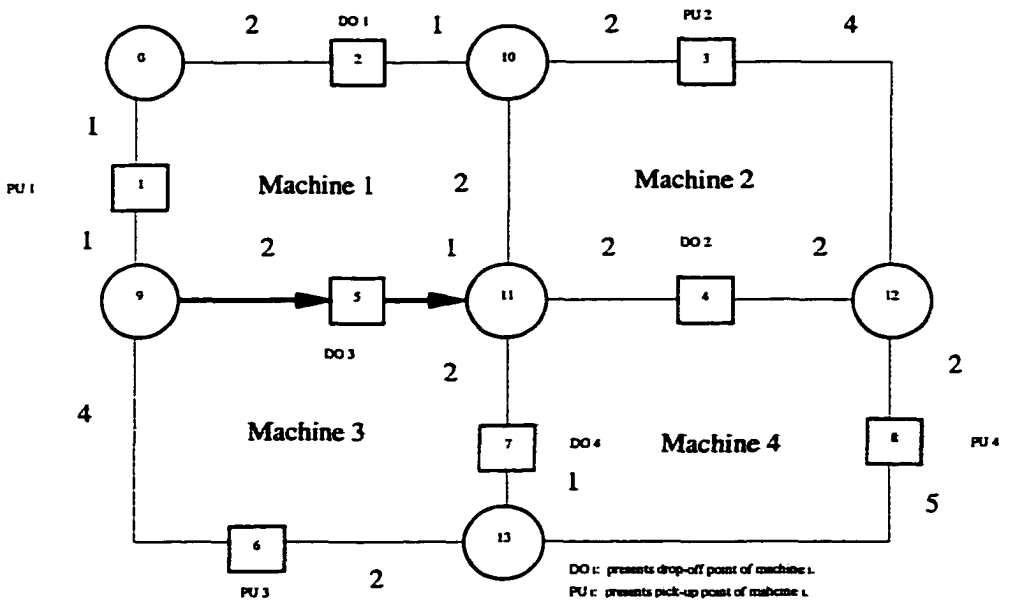


Figure 33. An example of a pre-processed network

Arc $_{9,5}$ and Arc $_{5,11}$ are fixed in the figure. The pre-processed network is ready for the application of the Complete algorithm, which is discussed next.

4.3.4 The Complete Algorithm

The pre-processed network is completed by applying the Complete algorithm. The required input data of the Complete algorithm are the pre-process network and the flow volume file. The output should be a feasible network in which any node on a flow path could be reached from any other node.

The Complete algorithm is presented as follows:

Step 1: Let K be the number of flows in the flow volume file and k be the priority index of flows.

Step 2: Set k = 1

Step 3: Initiate *S*, a set of arcs with fixed direction. That is, set $S = \phi$.

Step 4: Find the shortest path for the flow with priority k on the current network

4.a. Use Dijkstra's Algorithm to find the shortest path for the flow with the priority k on the current network.

4.b. If the shortest path includes at least one heavy arc, then try to find alternative paths with the same shortest distance by inactivating one of the heavy arcs in the path already found.

4.b.1 If the alternatives have fewer heavy arcs, then choose the one with the fewest heavy arcs as the shortest path of the flow.

4.b.2 Otherwise, if no alternatives exist or Step (4.b.1) is not applied, then use the original shortest path as the shortest path of the flow.

4.c. Expand the shortest path.

4.c.1. If the Total-degree of the first node in the path equals 2, expand the shortest path to include the arc that leads to the first node.

4.c.2. If the Total-degree of the last node in the path equals 2, expand the shortest path to include the arc that leads away from the last node.

4.c.3. If Step (4.c.1) produces a new first node that has a Total-degree of 2, repeat Step (4.c.1) until a new first node with a Total-degree of 3 or more is obtained.

4.c.4. If Step (4.c.2) produces a new last node that has a Total-degree of 2, repeat Step (4.c.2) until a new last node with a Total-degree of 3 or more is obtained.

4.d. Store all arcs in the shortest path in *S*.

Step 5: Fix the direction of arcs in *S* on the current network.

Step 6: If k = K, then go to Step 8. Otherwise, set k = k + 1 and go to Step 4.

Step 7: Evaluate the current network. That is, each node on a flow path should have at least one input arc and one output arc on the current network.

Step 8: The network is completed. Note that even though some edges and/or nodes are not used, the network obtained is feasible.

After being processed by the Complete algorithm, the pre-processed network in Figure 33 is completed as shown in Figure 34. In the figure, every node on a flow path has at least one

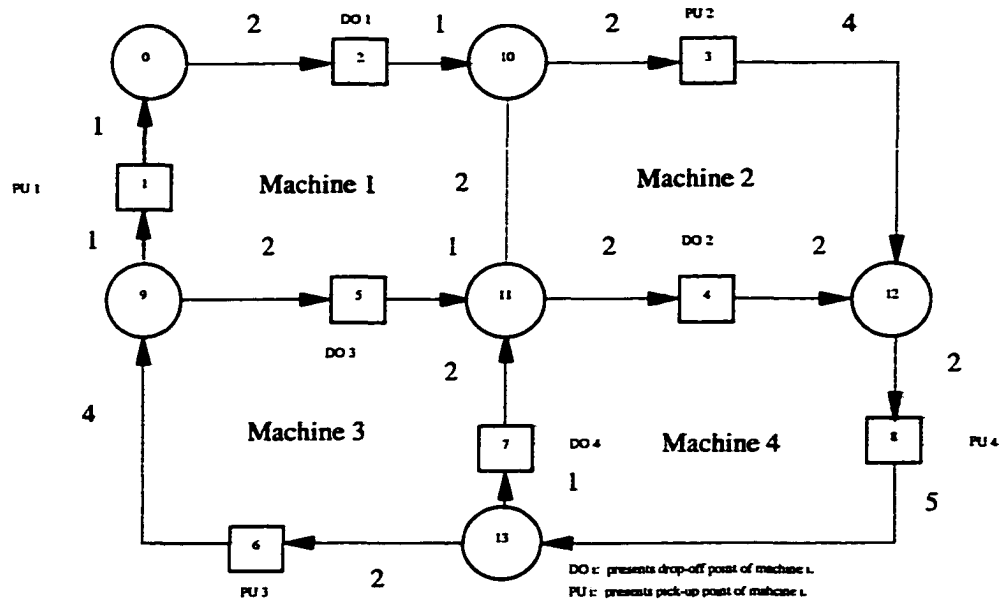


Figure 34. An example of a completed network

input arc and one output arc and the completed network is therefore feasible. Note that the edge bounded by nodes 10 and 11 is not used in this example.

4.3.5 The Pairwise-Interchange Algorithm

To swap the priorities of a pair of flows one at a time, the study employs the Pairwise-Interchange algorithm. In fact, the concept of the Pairwise-Interchange algorithm is embedded in the proposed AGV guidepath design procedure. The Pairwise-Interchange algorithm used in this study is briefly described as follows:

Step 1. Let K be the number of flows in the flow volume file, k be the priority index of the flow k, and j be the priority index of the flow j.

Step 2. Set the Total_Flow_distance = ∞

Step 3. Set k = 1

Step 4. Initiate j; that is, set j = 0

Step 5. 5.a. Invoke the Pre-process algorithm.

5.b. Invoke the Complete algorithm.

5.c. Compute the Total_Flow_Distance associated with the resulting completed network.

Step 6. Update the best network to the one with the minimum Total_Flow_Distance.

Step 7. 7.a. If $j = 0$, set $j = k + 1$ and go to Step 9.

7.b. If $1 \leq j \leq K\text{-}1$, set $j = j + 1$ and go to Step 9.

7.c. If $j = K$, go to Step 8.

Step 8. 8.a. If $1 \leq k \leq K\text{-}2$, then

8.a.1 Set $k = k + 1$

8.a.2 Set $j = k + 1$

8.a.3 Go to Step 9

8.b. If $k = K - 1$, then go to Step 11.

Step 9. Recover the original priority of flows in the flow volume file.

Step 10. Set the priority k to the flow j and the priority j to the flow k; go to Step 5.

Step 11. Terminate the procedure and output the best network with the minimum Total_Flow_Distance.

For an illustration, consider the flow volume file in Table 9. Originally, the priorities of flows are assigned according to their volume of flows, so that the flow with the largest volume has the highest priority. Suppose the first and the fifth flows exchange their priorities relative to each other. Then, the first flow will have the 5^{th} priority and the fifth flow will have the 1^{st} priority in the flow volume file, as shown in Table 10. Furthermore, according to the algorithm, the procedure will be executed $\frac{K(K-1)}{2}$ times.

4.4 The Proposed AGV Guidepath Network Design Algorithm

The guidepath design involves converting an undirected network to a directed network. Thus, the design process involves making decisions on what should be the flow direction on each undirected network segment between two nodes. An undirected segment becomes an arc when it is assigned a flow direction. The proposed procedure consists of three

Table 10. An example of interchanging

Flow #	From Machine #	To Machine #	Volume	Priority
1	**2**	**1**	**835**	**5**
2	3	2	780	2
3	1	3	777	3
4	3	4	558	4
5	**2**	**4**	**545**	**1**
6	4	1	389	6

components, the Pre-process algorithm, the Complete algorithm, and the improvement stage. The improvement stage is carried out using pairwise interchange.

Among the three components, the Complete algorithm is the core procedure. The Complete algorithm searches and fixes the shortest path for the flow with the highest priority in a flow volume file. This procedure is then repeated for the next highest priority flow and continued until the network is completed or until no more flow remains unconsidered. The shortest path of a flow on the network is heavily dependent on the shortest paths of higher priority flows, because the shortest path of a flow cannot be designed to violate the paths of other flows that have higher priorities. Each time a flow path is fixed, the succeeding flows must consider these earlier paths as constraints that cannot be violated.

Based on the Complete algorithm, the flows with higher production volume dominate the entire network construction procedure because of their higher priorities. Although the Complete algorithm is able to produce a feasible network, two things need to be considered. The first consideration is with regard to the process of fixing the direction of segments. The direction of a segment in a network is fixed because a flow with a higher priority uses the segment in its shortest path. However, it is possible that the sum of production volumes of some other flows with lower priorities may be larger than the production volume of the flow with a higher priority. In the meantime, these flows with lower priorities require that the segment be set in the other direction. Thus, a better network that has a smaller total flow

distance could be neglected. To avoid the situation, the Pre-process algorithm is employed in the proposed AGV guidepath design algorithm. In the Pre-process algorithm, the situation is improved by evaluating the flow volume for each arc individually, after which a pre-processed network is produced. As a result, the flows with lower priorities might benefit from the Pre-process algorithm.

The other consideration is with regard to the priority of flows. Originally, the flows are arranged in a non-increasing order based on their size. According to the order, the priorities of flows are assigned. However, when the production volumes of some flows are tied, or the differences between the production volumes of flows are negligible, the priority order of flows might prevent the design of a better network. To reduce this possibility, the concept of the Pairwise-Interchange algorithm is embedded in the proposed AGV guidepath design procedure. Through use of this concept, the priorities of pairs of flows are swapped one at a time. Then, the flow volume file with the swapped priorities is employed to obtain a new network design. If the new network design yields an improvement, the current best network is updated. If no improvement occurs, a new interchange is generated and used to design an associated network. The procedure is continued until no further improvement is observed; then the best network design with the minimum total flow distance is generated. The proposed AGV guidepath design algorithm can be represented by a flow chart, as shown in Figure 35.

The proposed AGV guidepath design algorithm is as follows:

Step 1: Input the distance matrix and the flow volume file.
Initiate a set, A. That is, store the flow volume file in A.
Initiate a set, B. That is, set $B = A$.

Step 2: In set A, let K be the number of flows, k be the priority index of the flow k, and j be the priority index of the flow j.

Step 3: Initiate the current best network, N. Initiate the total flow distance associated with the best network, N. That is, set Total_Flow_Distance = ∞.

Step 4: Set k = 1

Step 5: Initiate j. That is, set j = 0

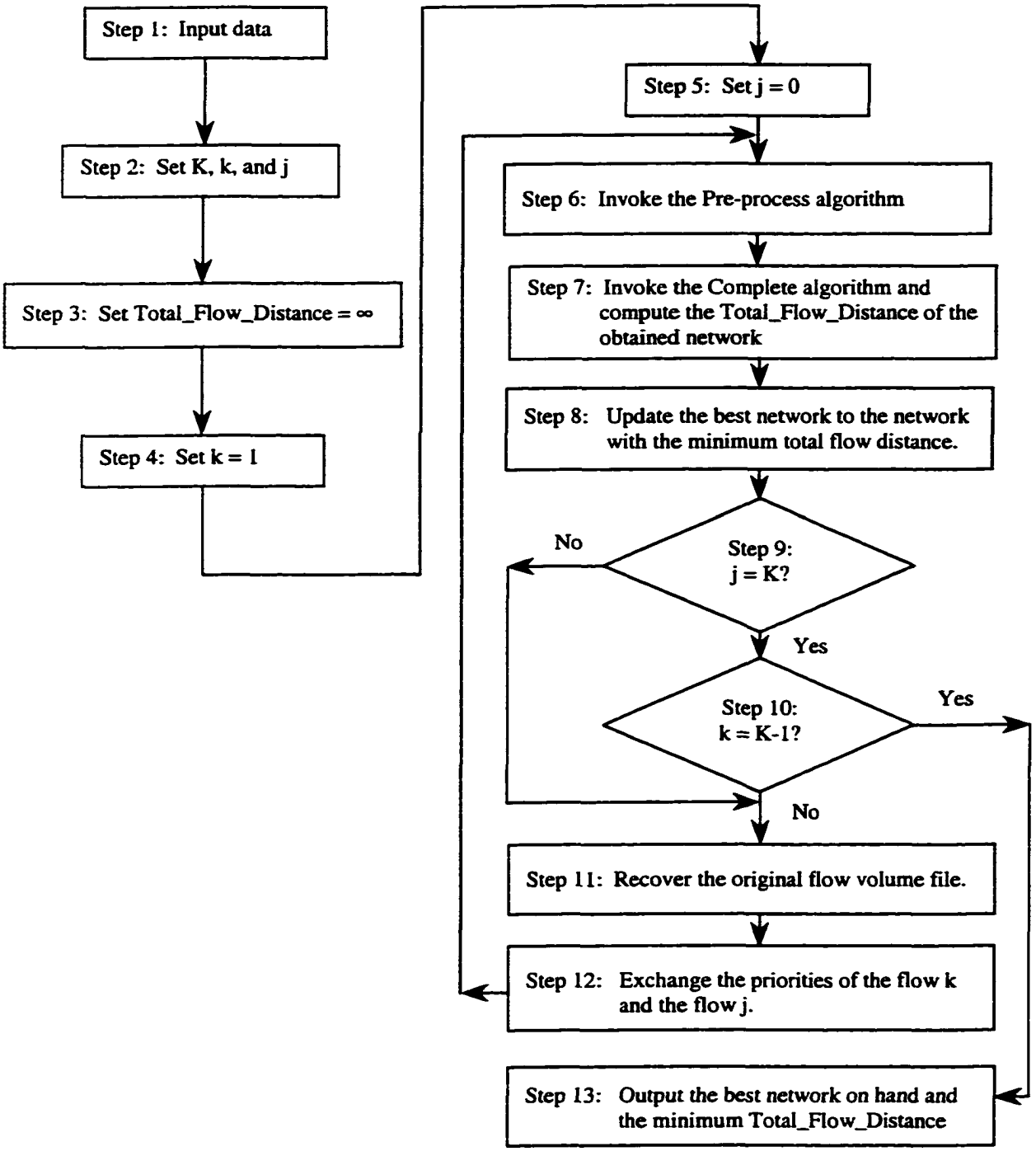


Figure 35. The flow chart of the proposed AGV guidepath design procedure

Step 6: Taking set B as input, pre-process the network by invoking the Pre-process algorithm.

Step 7: Taking set B as input, complete the network by invoking the Complete algorithm and return the total flow distance of the resulting network. Let the returned total flow distance equal TF.

Step 8: 8.a. If TF < Total_Flow_Distance, then,

8.a.1 Set Total_Flow_Distance = TF

8.a.2 Update the current best network, N, to the network that yielded TF and go to Step 9.

8.b. Otherwise, go to Step 9.

Step 9: 9.a. If $j = 0$, set $j = k + 1$ and go to Step 11.

9.b. If $1 \leq j \leq K\text{-}1$, set $j = j + 1$ and go to Step 11.

9.c. If $j = K$, go to Step 10.

Step 10: 10.a. If $1 \leq k \leq K\text{-}2$, then

11.a.1 Set $k = k + 1$

11.a.2 Set $j = k + 1$

11.a.3 Go to Step 11

10.b. If $k = K - 1$, then go to Step 13.

Step 11: Set $\boldsymbol{B} = \boldsymbol{A}$.

Step 12: In set $\boldsymbol{B}$, switch the priority of the k^{th} flow to the priority of the j^{th} flow and the priority of the j^{th} flow to the priority of the k^{th} flow. Go to Step 6.

Step 13: Terminate the procedure and output the best network, N, and the associated total flow distance, Total_Flow_Distance.

For the sake of illustration, the proposed AGV guidepath design procedure is tested by using the distance matrix of Table 8 and the flow volume of Table 9. The best network obtained is as shown in Figure 36. Note that no direction is assigned to the edge bounded by node 10 and node 11, because no flows passed through either Arc $_{10,11}$ or Arc $_{11,10}$. The total flow distance is equal to 23101 distance units, which is exactly the same as was reported [27].

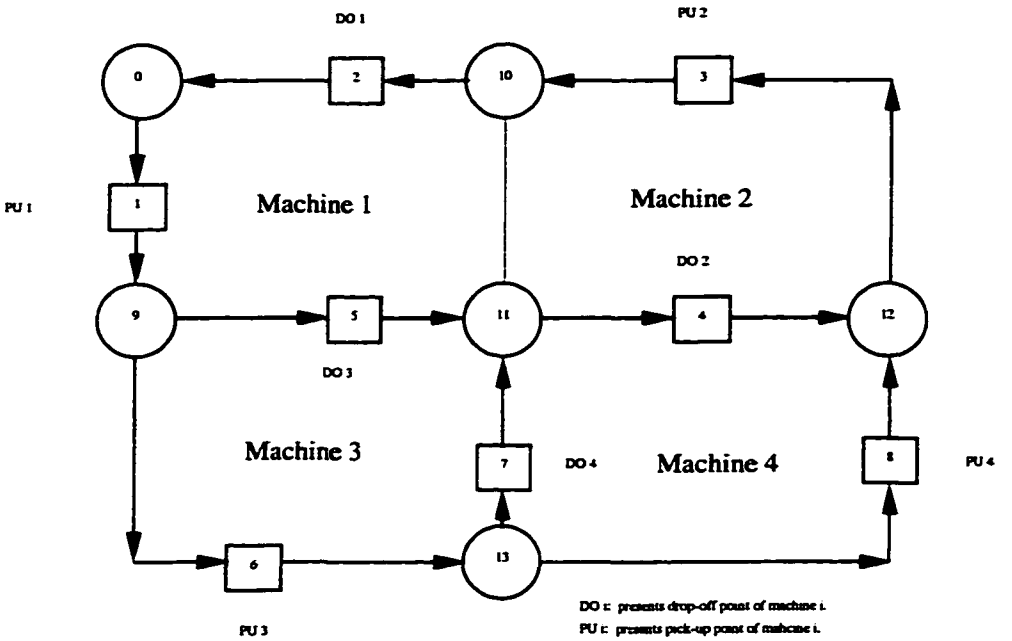


Figure 36. The best network obtained by the proposed procedure

4.5 Test Results

To evaluate the performance of the proposed algorithm, two more examples drawn from the literature are employed as a test. The first example is taken from Kaspi and Tanchoco's paper [24]. The associated flow volume file is shown in Table 11, the network generated by their approach is shown in Figure 37, and the network obtained by using the proposed procedure is shown in Figure 38. The total flow distance is equal to 10165 distance units, an improvement over the 10170 distance units obtained by Kaspi and Tanchoco [24]. The differences between the two networks are marked by the use of dotted lines in Figure 37 to highlights Arc $_{11,2}$, Arc $_{2,13}$, Arc $_{13,14}$, Arc $_{14,3}$, Arc $_{3,23}$, and the edge bounded by nodes 10 and 18. In addition, the computational time of 90 seconds to solve the problem is significantly lower than the 27.5 minutes reported by Kaspi and Tanchoco [24]. However, it should be noted that the computer environments used in the two studies might be different. The proposed procedure was implemented on a DEC Alpha workstation running of 300MHZ.

Table 11. A flow volume file [24]

Flow #	From node #	To node #	Volume	Priority
1	1	7	50	1
2	1	4	30	2
3	5	8	30	3
4	6	8	30	4
5	7	8	30	5
6	7	5	25	6
7	1	2	20	7
8	1	9	20	8
9	4	5	20	9
10	5	7	20	10
11	7	6	15	11
12	2	3	10	12
13	3	8	10	13
14	3	9	10	14
15	4	6	10	15
16	9	3	10	16
17	9	4	10	17
18	9	6	10	18
19	2	6	5	19
20	2	9	5	20
21	9	5	5	21

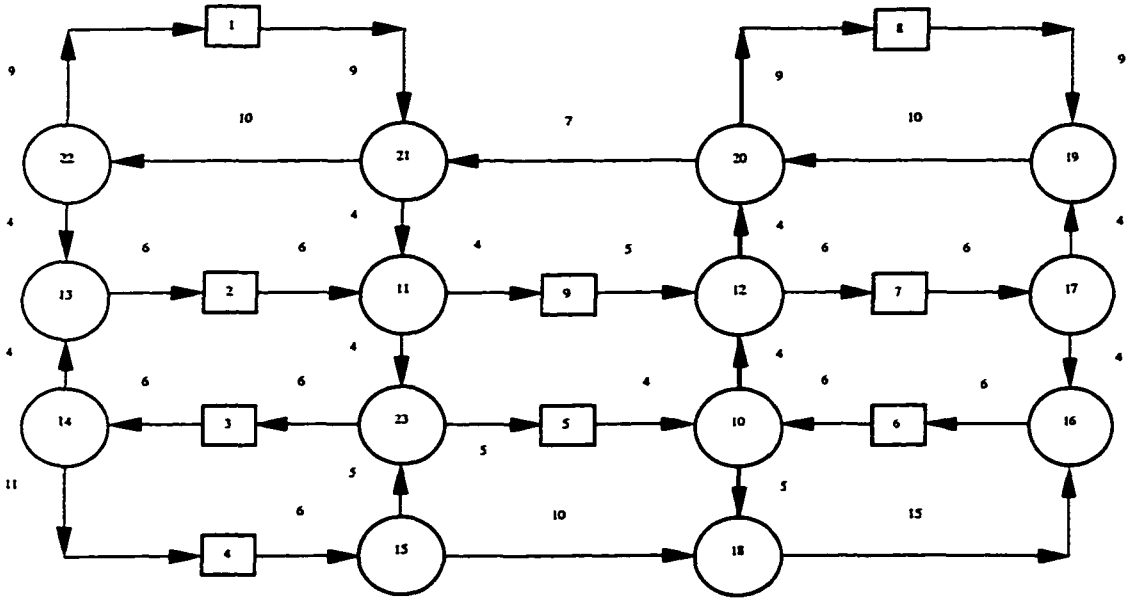


Figure 37. The network obtained in [24]

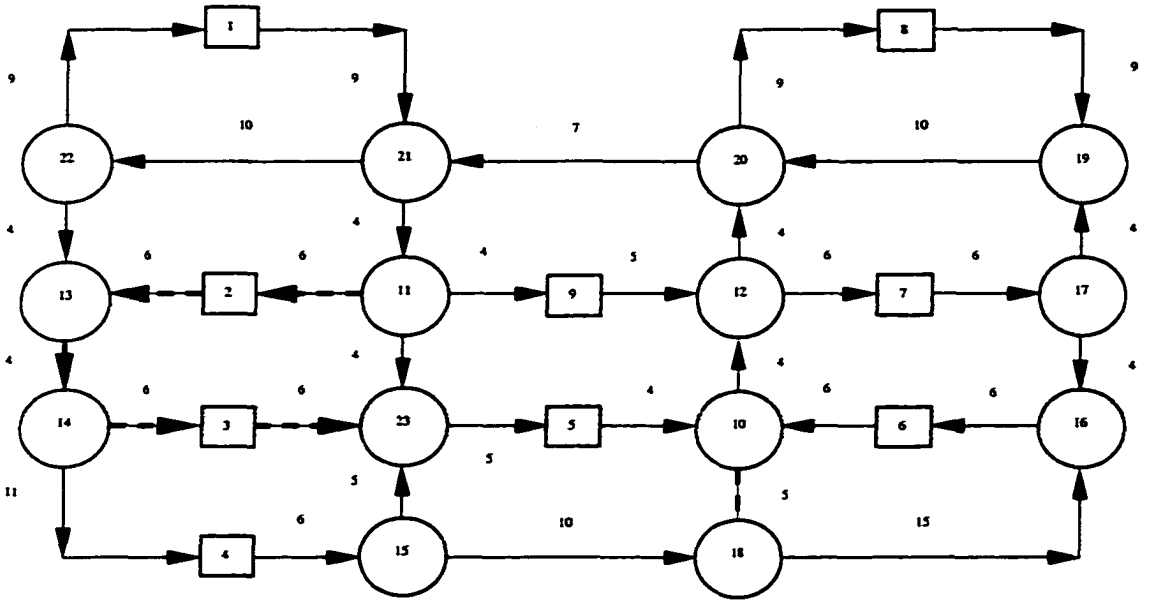


Figure 38. The network obtained by using the proposed procedure

In the second example, which is smaller and which is taken from a published paper [23], the associated flow volume file is given in Table 12 and the network is shown in Figure 39. The network produced by using the proposed procedure is exactly the same as in Figure 39. The total flow distance is 12400 distance units, and the time taken to generate the solution is 5 seconds.

4.6 Discussion

AGV guidepath network design procedure has been presented in this chapter. The performance of the algorithm was tested by using some examples from the literature. The networks generated by the algorithm were equal to or better than the published results. Furthermore, the computational time required by the procedure is far less than that of its competitors. The AGV guidepath network design procedure is robust and able to produce a feasible network within a reasonable time.

Table 12. The flow volume file in [23]

Flow #	From machine #	To machine #	Volume	Priority
1	2	3	100	1
2	3	1	80	2
3	1	2	70	3
4	1	3	70	4
5	2	1	50	5
6	3	2	30	6

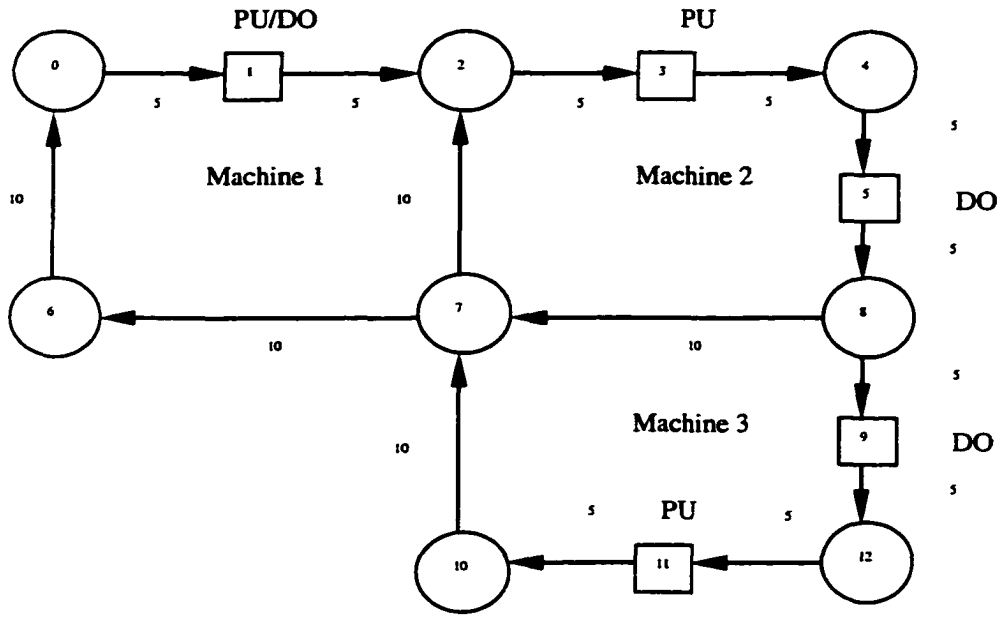


Figure 39. The obtained network, the same as in [23]

In a virtual production system, the networking module performs the machine connection and plays a role in linking together machines belonging to the same machine cell. With the AGV guidepath network design procedure developed, the networking module can be updated in response to a changing product mix, thereby minimizing the material handling distance.

CHAPTER 5. A VIRTUAL CELLULAR MANUFACTURING SYSTEM

In the processing system configuration module of a virtual production system, virtual cellular configuration is one possible production configuration. A shop with virtual cellular configuration and an associated virtual AGV guidepath network is a complete virtual cellular manufacturing system in the strong sense. If either the production system or the network system but not both is virtual, the entire system can be termed a virtual system in a partial sense. Combinations of the production systems design and network design generate multiple variations of virtual production systems, as identified in Chapter 1. In Chapters 3 and 4, the Ko's virtual cell formation procedure and the AGV guidepath network design procedure were presented. These procedures for constructing a virtual cellular manufacturing system and virtual guidepath provide the basis for constructing various virtual production systems. However, before the construction of a virtual production system is discussed, a virtual cellular manufacturing system is first described in this chapter.

For the sake of illustration, consider the job routings in Figure 14 and the associated shop layout in Figure 15. The related flow volume file for the data in Figure 14 could be provided as shown in Table 13. If each workstation (machine) in Figure 15 occupies a 10 by 10 block on the shop floor and has the pick-up point and the drop-off point at the same location, the shop could be represented by a set of nodes and undirected arcs, as shown in Figure 40.

Application of the developed AGV guidepath network design algorithm to interconnect the machines in each virtual cell, as shown in Figure 18, yielded the directed virtual network of Figure 40. The network of Figure 41 is based on the workstation layout of Figure 19. The AGV network links together machines or workstations belonging to the same cell. The network construction tends to minimize the total material handling time. Based on the material flow requirement between cells as well as within cells, the total travel time for the guidepath of Figure 40 is 45970 distance units. It takes 4.5 minutes on DEC Alpha workstation running 300MHZ to solve the model. The machine cell network for each of the six virtual cells in Figure 19 and later in Figure 41 might be individually presented as shown in Figures 42 - 46.

Table 13. The related flow volume file in Figure 13

No.	From node #	To node #	Flow volume	Priority
1	7	8	930	1
2	8	5	930	2
3	9	7	930	3
4	13	14	880	4
5	14	15	880	5
6	15	16	880	6
7	16	17	880	7
8	1	2	755	8
9	2	3	755	9
10	4	18	605	10
11	5	4	605	11
12	5	6	605	12
13	6	10	605	13
14	18	5	605	14
15	17	1	555	15
16	7	13	475	16
17	10	12	475	17
18	11	10	475	18
19	12	7	475	19
20	10	13	405	20
21	5	11	325	21
22	10	1	200	22

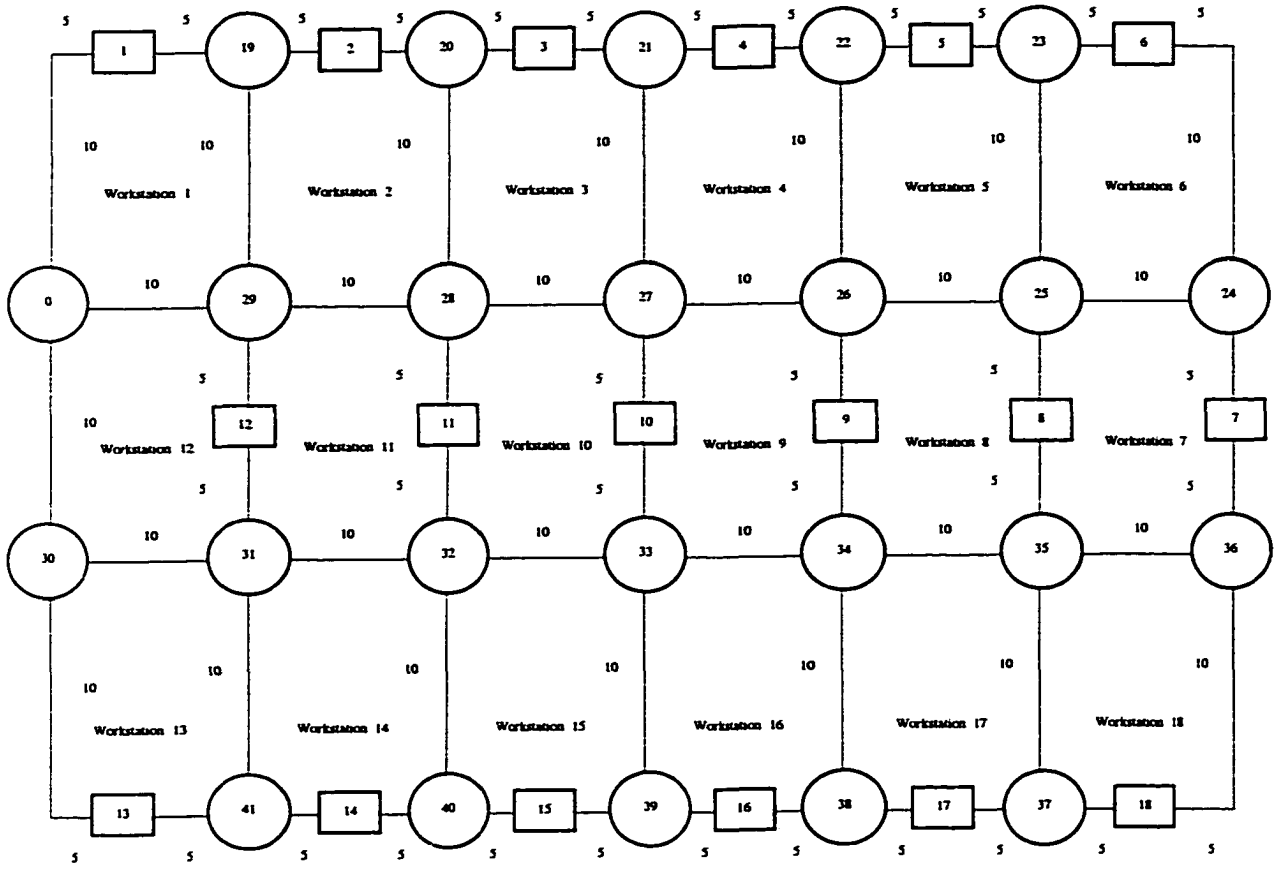


Figure 40. The shop in Figure 15, represented by a set of nodes and undirected arcs

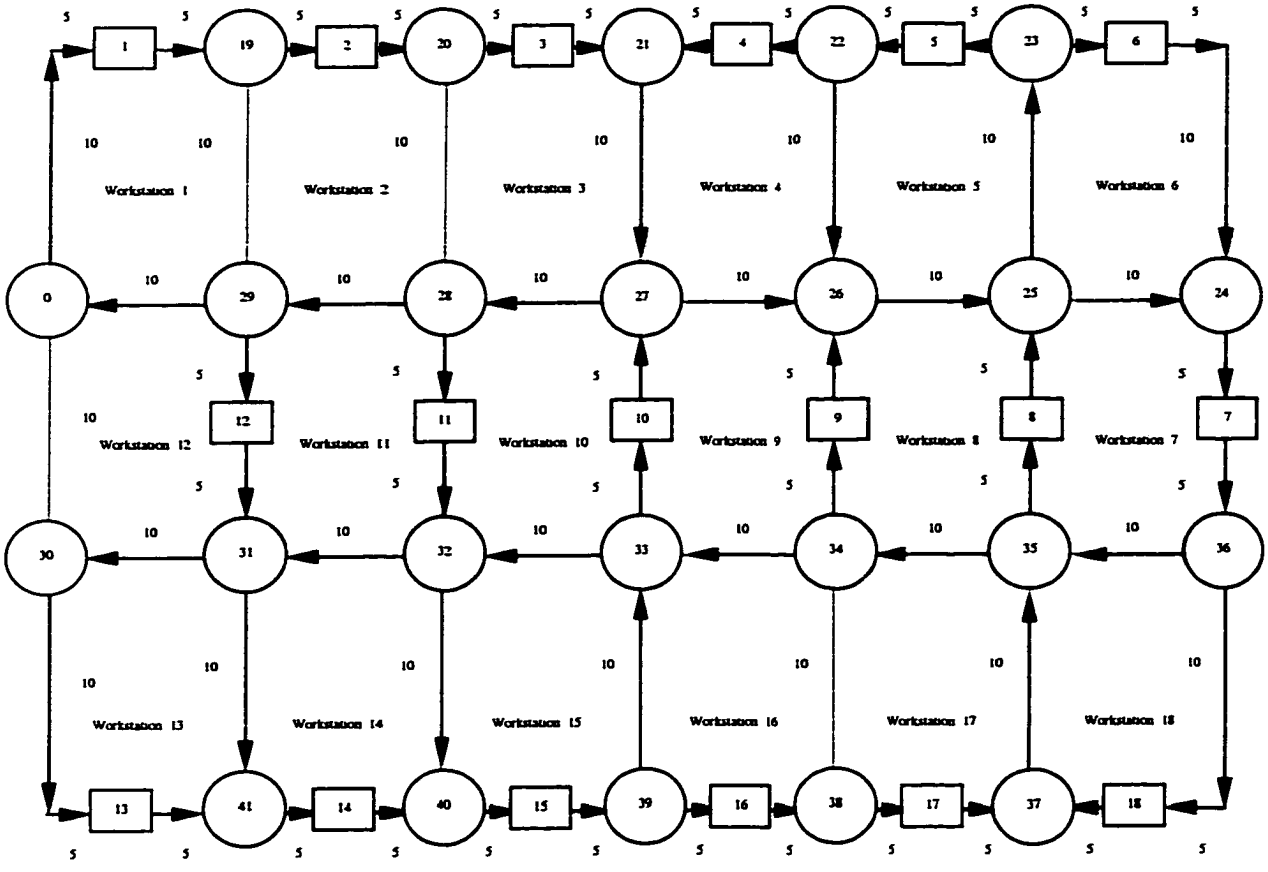


Figure 41. The network generated by the proposed AGV guidepath design procedure

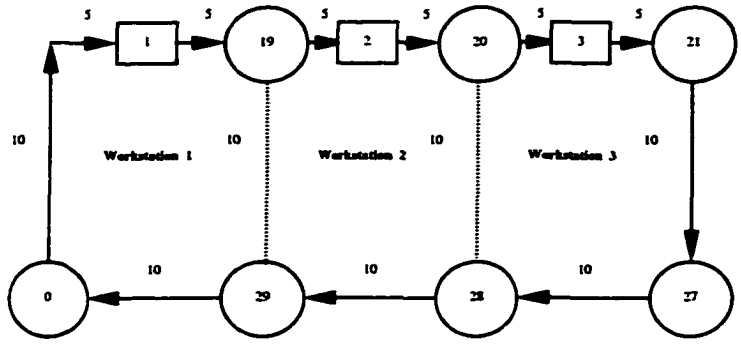


Figure 42. The virtual AGV network of virtual cell 1 consisting of workstations 1, 2, and 3

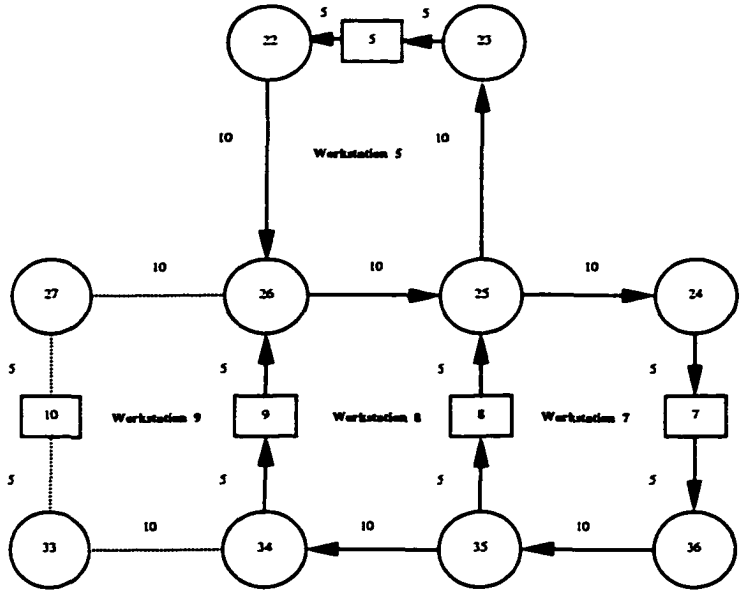


Figure 43. The virtual AGV network of virtual cell 2 consisting of workstations 9, 7, 8 and 5

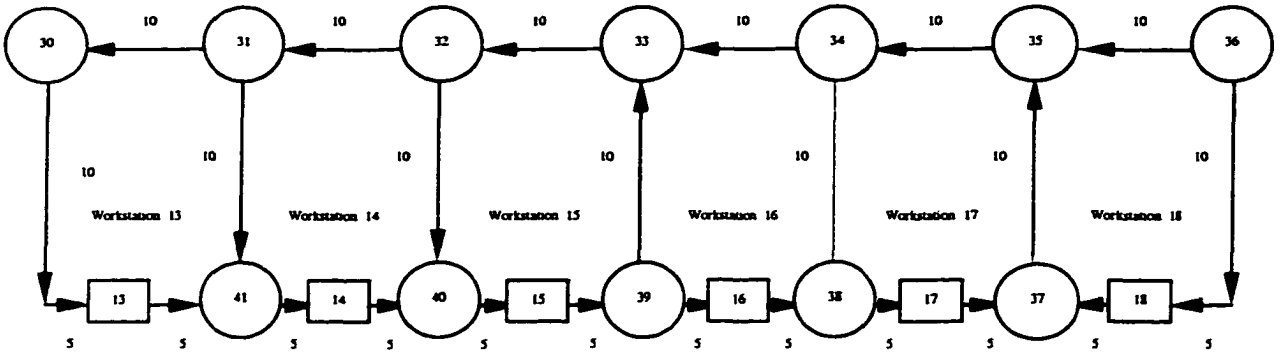


Figure 44. The virtual AGV network of virtual cell 3
consisting of workstations 13, 14, 15, 16, and 17

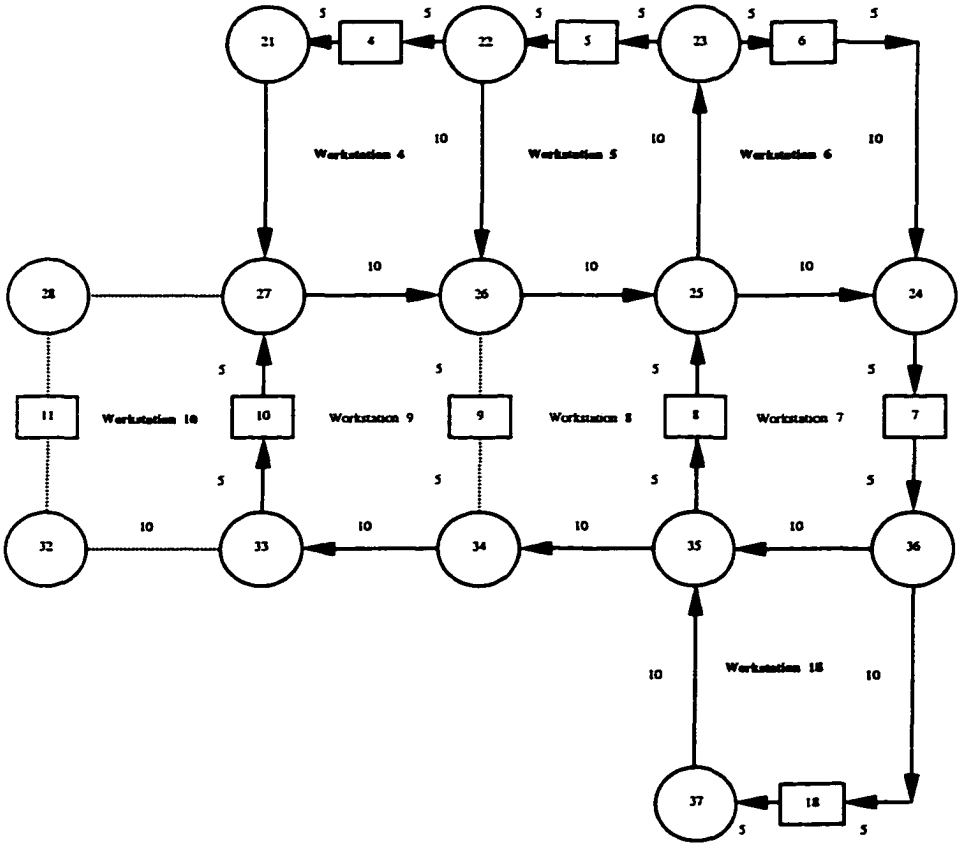


Figure 45. The virtual AGV network of virtual cell 4,
consisting of workstations 4, 5, 6, 10, and 18

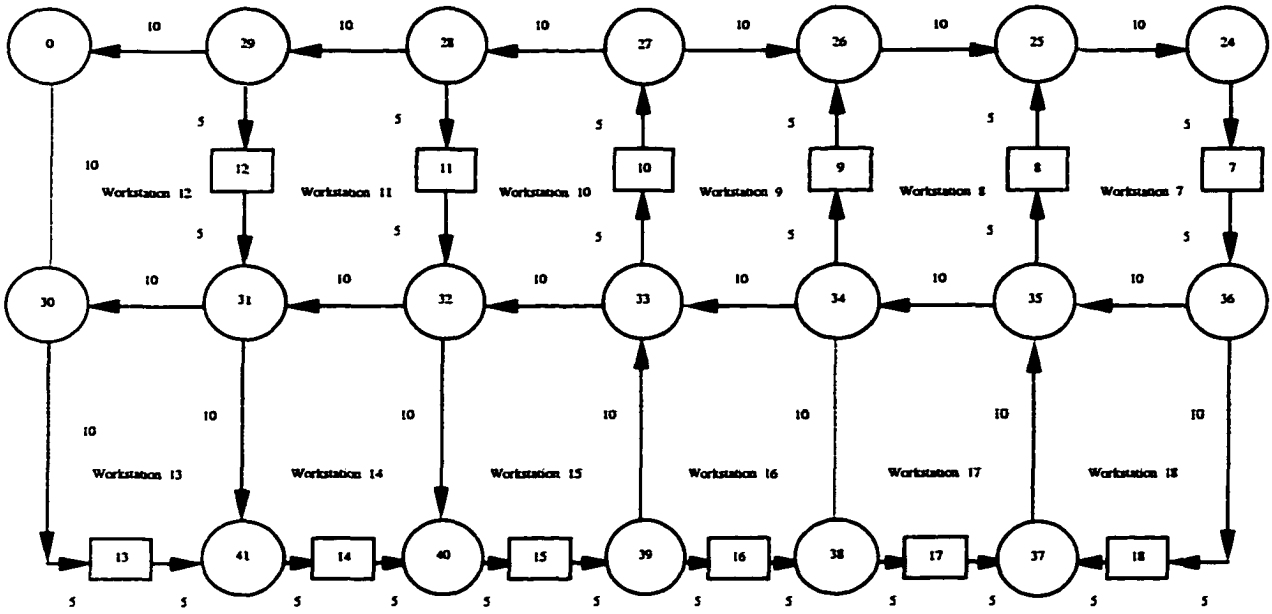


Figure 46. The virtual AGV network of virtual cell 5, consisting of workstations 11, 10, 7, and 12

As shown in this chapter, a job shop is converted into a virtual cellular manufacturing system. First, virtual cells were formed by using the developed virtual cell formation procedure. Then, machines belonging to the same virtual cell were linked together by using the proposed AGV guidepath network design procedure. As shown in this chapter, with the two proposed procedures, a job shop type plant layout is converted into a virtual cellular manufacturing system.

A virtual production system consists of a processing system configuration module and a networking module. The processing system configuration module determines the best production configuration for a shop. The networking module helps redesign an associated AGV guidepath network in order to minimize the material handling distance. A virtual production system provides an alternative in operating a shop in which physical relocation of the machines is impractical or infeasible.

CHAPTER 6. VIRTUAL PRODUCTION SYSTEM

In this research, application of the virtual production system concept is presented as an alternative approach to operate a factory that was initially designed as job shop but that later encountered a changing product mix. Sample production data are generated, planned for, and analyzed in this chapter, and the performances of alternative production system designs are compared.

6.1 Introduction

The production environment examined in this research is a job shop in which the machines or workstations cannot be physically rearranged to obtain a more efficient layout because of the nature of the machines. The material handling tasks are performed by free ranging AGVs in the shop; this makes it possible to reconfigure the AGV flow network quite readily as needed. The shop experiences a changing product mix, which points out the shortcoming of employing the fixed machine grouping and AGV guidepath layout. Application of the virtual production system concept is to be used to allow the shop to adapt its operation to the changing product mix. The focus of this research is to develop an algorithmic procedure for constructing such a virtual production system, using the existing job shop and the routing of the product as inputs to the algorithms, which can be used during each production session to select the best production system configuration.

The four types of virtual production system were described in Section 1.2.3. In addition, two traditional production system designs are considered in this research: the job shop and the traditional cellular manufacturing system. During any production session, one of these six production system types can be selected to operate the shop. The production system type that best adapts to the product mix on hand is selected. The objective is to operate the shop under the most efficient production situation during each production session as the product mix changes from one session to another.

In this chapter, the relative performances of production systems using the same set of product mix and production resources are compared. The experimental design for making the comparisons is described in the following section.

6.2 Experimental Design

One form of production system is selected over competing alternatives for a given product mix on the basis of performance of the shop. If the target shop performs better under production system type β than under any other production systems, with a given product mix and resource level, then production system type β is selected and the target shop is then operated as a production system type β for that particular instance. To arrive at a decision regarding performance, the expected performance under each production system scenario would be computed and compared with performance under other competing production system scenarios. In this chapter, the experimental design used in making the comparisons is illustrated with some examples. The measures of performance are machine setup time, material handling time, and weighted performance value.

The experiment consists of five examples, each of which covers ten production sessions. Each example has a fixed number of machines: 12, 8, 5, 11, and 8 machines in Examples 1, 2, 3, 4, and 5, respectively. The product mix data, except for Example 1, which is modified from the literature, is generated randomly. To create a product mix for each production session, three parameters need to be given: the number of machines in the shop, the number of parts to be completed, and the largest length of the part routings (in terms of machines). With the three parameters, the product mix data is then created randomly. As shown in Table 14, for Production Session 2 in Example 2, the values for the number of machines, the number of part routings, and the largest length of part routings are arbitrarily assigned the values 8, 12, and 5, respectively. The created product mix data is presented in Appendix D.

The jobs in a production session of each example can be further classified into several part families/groups; for instance, the jobs of Production Session 1 in Example 1 can be divided into three groups (Groups 1, 2, and 3), as shown in Appendix D. Note that the product mix changes as the production session changes.

For each example, it is assumed that a corresponding basic job shop layout exists, as shown in Figures 47-51. It is also assumed that the corresponding AGV guidepath network exists. However, to operate the shop as any production system other than a job shop, the given production data would have to be analyzed and subjected to the design procedures of

Table 14. The parameters used to generate the product mix data

Example	Production Session	The number of machines	The number of part routings	The largest length
Example 2	1	8	11	4
	2	8	12	5
	3	8	8	4
	4	8	13	5
	5	8	10	3
	6	8	7	4
	7	8	15	5
	8	8	9	4
	9	8	11	4
	10	8	15	5
Example 3	1	5	7	5
	2	5	10	4
	3	5	12	5
	4	5	9	4
	5	5	16	5
	6	5	11	4
	7	5	15	6
	8	5	12	5
	9	5	10	4
	10	5	14	5
Example 4	1	11	9	5
	2	11	9	5
	3	11	12	4
	4	11	14	4
	5	11	8	5
	6	11	10	5
	7	11	8	6
	8	11	18	6
	9	11	10	4
	10	11	9	5
Example 5	1	8	10	4
	2	8	12	4
	3	8	8	6
	4	8	10	4
	5	8	10	4
	6	8	12	5
	7	8	11	4
	8	8	5	6
	9	8	9	4
	10	8	14	4

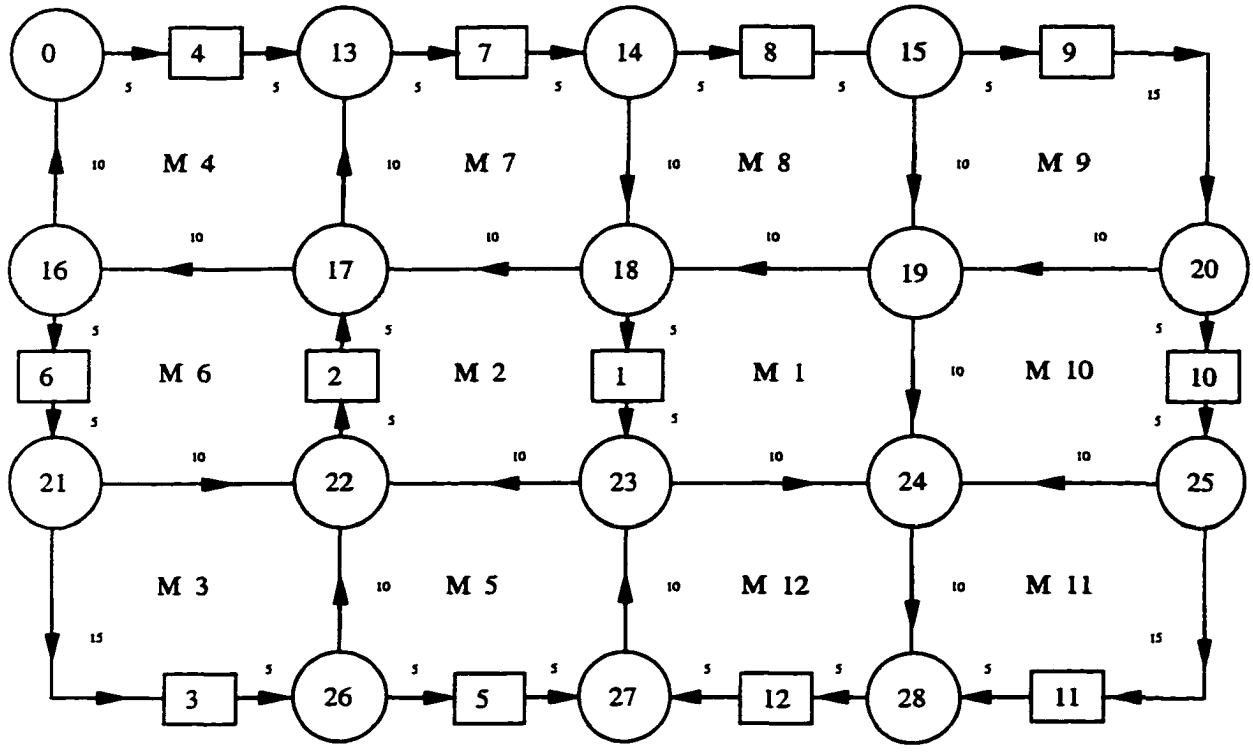


Figure 47. The layout and network of Example 1 as it exists as a job shop

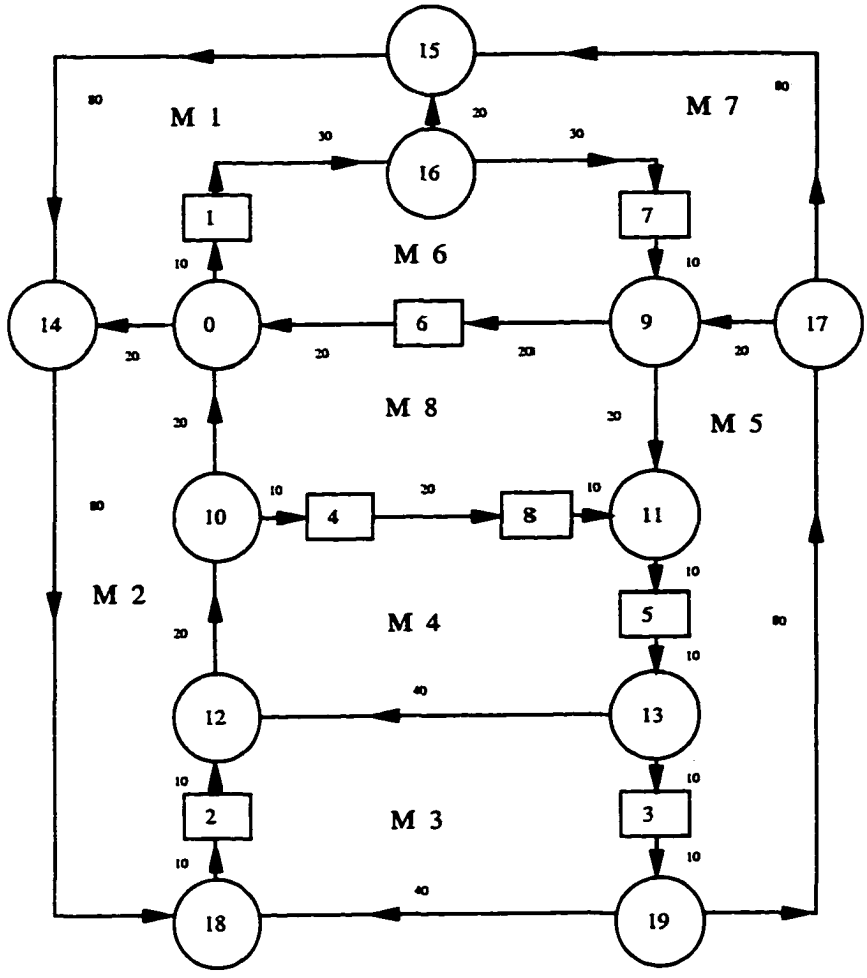


Figure 48. The layout and network of Example 2
as it exists as a job shop

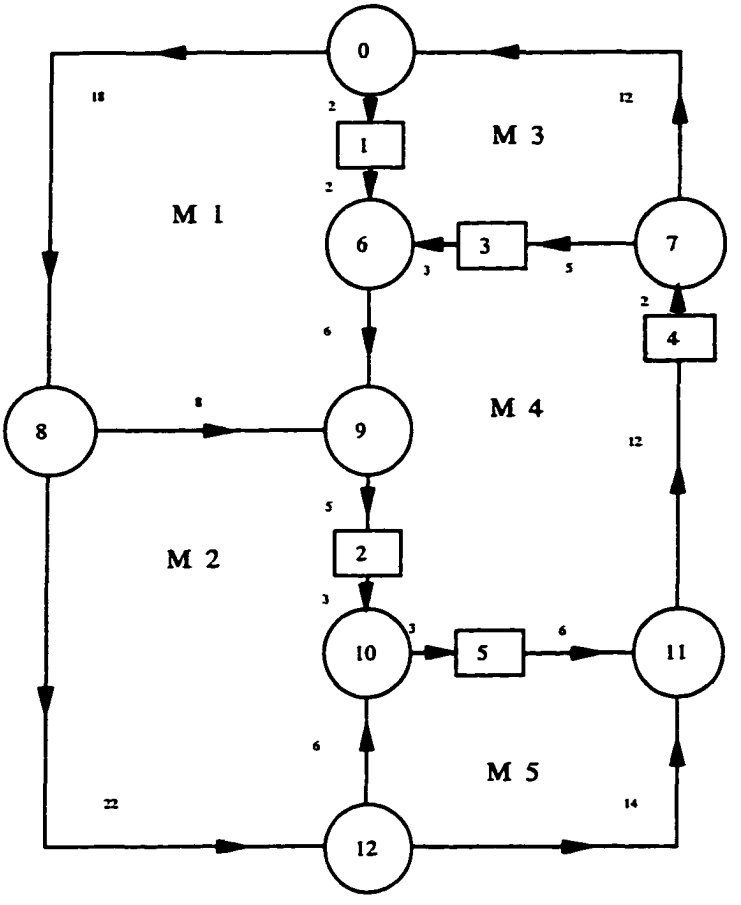


Figure 49. The layout and network of Example 3 as it exists as a job shop

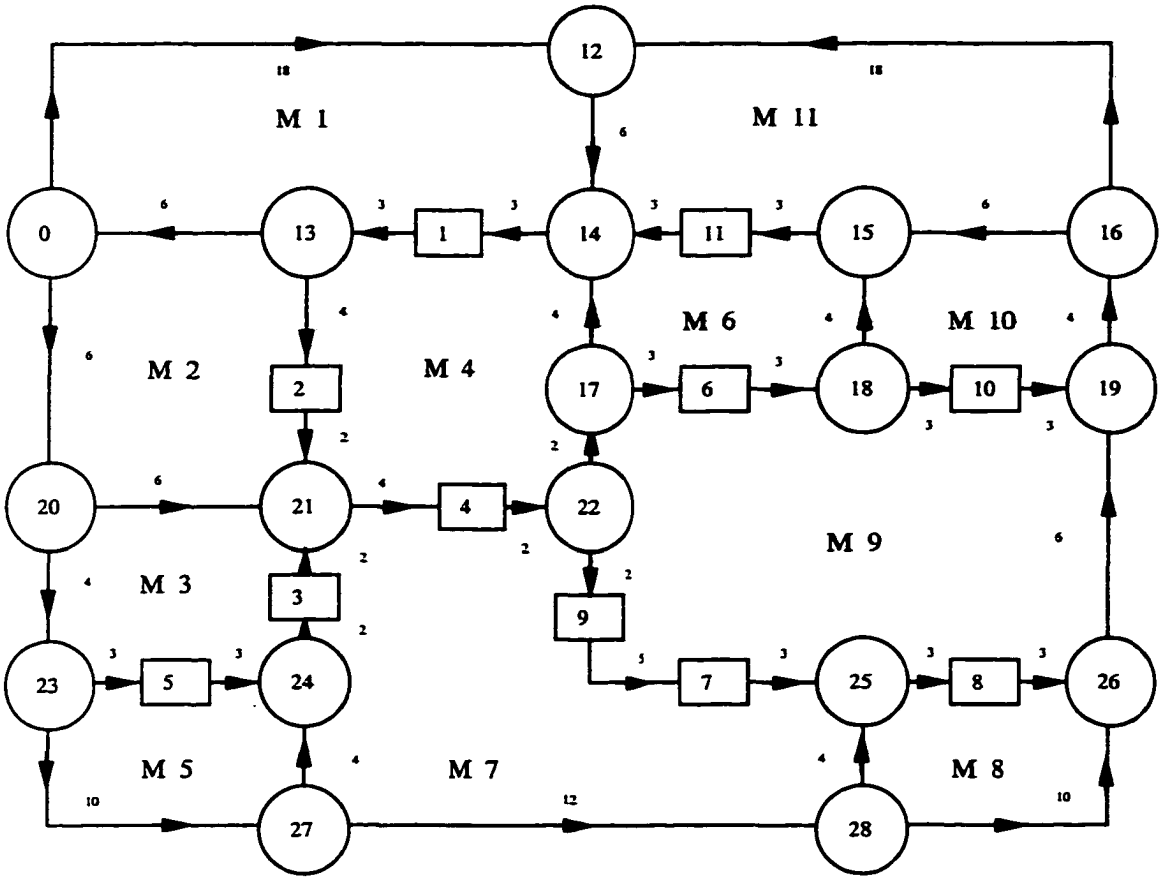


Figure 50. The layout and network of Example 4 as it exists as a job shop

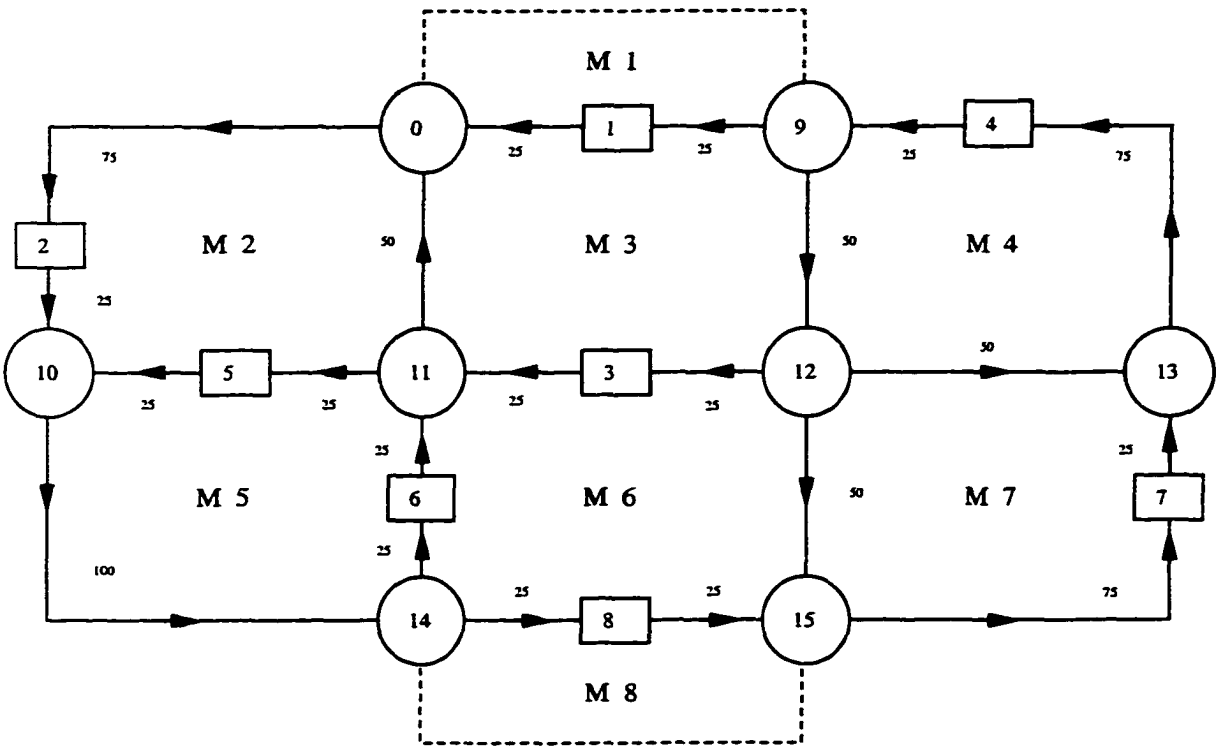


Figure 51. The layout and network of Example 5
as it exists as a job shop

the processing system configuration module and the networking module, as described in Chapters 3 and 4. The formation of the other production system alternatives using the data provided is illustrated in the sections that follow.

6.2.1 The Construction of Traditional Cellular Manufacturing

With the default layouts for the five examples (Figures 47-51), the construction of traditional cellular manufacturing systems is subject to how machines are grouped into machine cells. Once designed, machine cells remain the same throughout the example. Machines belonging to the same machine cell are linked by AGVs using the given flow network for the example.

To form machine cells for the five examples, a mathematical model from the literature [44] and Section 2.1.2 is employed. In this technique, the number of machine cells must be assigned artificially. The numbers of machine cells are specified as 3, 2, 1, 3, and 2 for Examples 1, 2, 3, 4, and 5, respectively. Cell formation for each example is based on the product mix of its first production session. Once machine cells are formed, the same cell configuration is used through the ten production sessions in an example. In forming machine cells, LINGO, a commercial mathematical software program, is employed, except for Example 3, which has only one machine cell. An example of the LINGO program (for Example 1) is as shown in Appendix E. The cell formations obtained for the five examples are summarized in Table 15; for instance, in Example 4, Machine Cell 2 consists of machines 6, 7, 10, and 11.

Table 15. The cell formations in traditional cellular manufacturing

	Machine		
Examples	**Machine Cell 1**	**Machine Cell 2**	**Machine Cell 3**
Example 1	1, 4, 7, 8, 9	2, 3, 5, 6	10, 11, 12
Example 2	1, 4, 5, 6, 8	2, 3, 7	
Example 3	1, 2, 3, 4, 5		
Example 4	3, 8, 9	**6, 7, 10, 11**	1, 2, 4, 5
Example 5	2, 6, 7, 8	1, 3, 4, 5	

6.2.2 The Construction of Virtual Production Systems

The layouts employed for the virtual production systems are the same as for the job shops (Figures 47-51). However, the associated AGV guidepath networks might be updated by the proposed AGV guidepath network design procedure (see Chapter 4) from one production session to another. Furthermore, although machines remain in their fixed locations as in the job shops, they are grouped or organized differently from those of the corresponding job shops.

Each of four types of virtual production system has its own characteristics, as described in Section 1.2.3. Type I (VC/FN) updates virtual cells in every production session, while the associated AGV guidepath network remains the same. Type II (JS/VN) works the same as a job shop, but the associated AGV guidepath network is updated as the product mix changes. Type III (MC/VN) uses the same cell configuration as traditional cellular manufacturing (in Section 6.2.1), but the associated AGV guidepath network is updated as the product mix changes. Type IV (VC/VN) can update both the virtual cells and the associated AGV guidepath network as the product mix changes.

With the product mix data in Appendix D, virtual cells generated by using the Ko's virtual cell formation procedure for each production session are shown in Appendix F. To illustrate, virtual cells in Production Session 1 of Example 1 are presented in Figure 52, in which five virtual cells (Virtual_Cells 1, 2, 3, 4, and 5) are created for the production session. As can be seen, Virtual_Cell 1 consists of machines 4, 7, 8, and 9. In addition, job routings can be represented by using virtual cells; for instance, Job 1 must sequentially visit Virtual_Cell 3 and Virtual _Cell 1 to be completed.

After the production systems are constructed, they are compared on the basis of their machine setup time and material handling distance requirements, as described in Sections 6.3 and 6.4, respectively. In addition, a composite performance value is computed, as presented in Section 6.5.

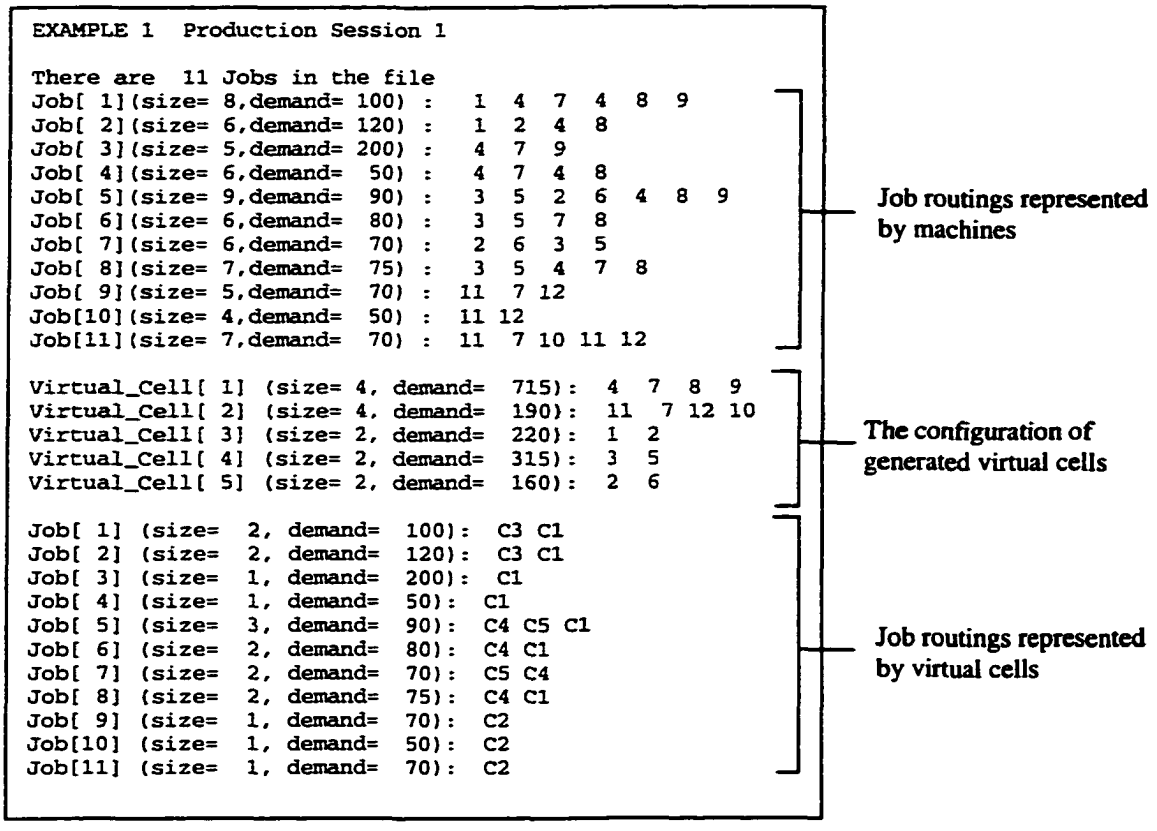


Figure 52. The virtual cell configuration in Production Session 1 of Example 1

6.3 Comparison of Setup Time

As described in Section 1.2.1 and Figure 6, three alternatives for machine arrangements in a shop are the traditional job shop configuration, traditional cellular configuration, and virtual cellular configuration. With different configurations, different setup times might be incurred, even with the same product mix.

The job shop configuration treats jobs individually and assumes that no relationship exists between jobs, so that a machine setup must change entirely between jobs. However, in traditional cellular and virtual cellular configurations, jobs are classified into part families/groups, based on shapes, required processing procedure, and so forth. Jobs belonging to the same part family/group share the same major setup, or a common setup can

be designed. A major change in machine setup is necessary only when a machine/virtual cell's production changes from one part family/group to another. Within the same part family/group, different jobs require only minor changes of machine setup, such as swapping a fixture or changing an NC code. Creation of related setup-time tables is described in the following section.

6.3.1 Creation of Setup-Time Tables

By use of Microsoft Excel, the required setup-time tables are created automatically. The setup-time tables needed for Production Session 1 in Example 1 are shown in Tables 16-55; setup-time tables for the other production sessions in the experiment are available from the author upon request.

The setup-time tables in Tables 16-27 are used for job shop configuration; each machine has a unique setup-time table. For example, Table 16 is the setup-time table for machine 1, which machine produces two jobs (Job 1 and Job 2). In a setup-time table, the setup time is randomly generated by RAND(6, 10), one of the functions provided in Microsoft Excel, in which a value between 6 and 10 is randomly produced by the function. For example, when the production of machine 1 switches from Job 1 to Job 2, the required setup time is 6 time units, as shown in Table 16.

Tables 28-40 show the major setup-time tables for traditional cellular configuration. As shown in Table 15, there are three machine cells for Example 1. Table 28 is the major setup-time table for the three machine cells. Empty slots in the table mean the job group will not visit the machine cell. The major setup time is computed by multiplying the number of machines required to process a part family/group in a machine cell and the number generated by RAND(6, 10). For example, when employment of Machine Cell 1 is changed from Group 1 to Group 2, a major setup time (32.1 time units) is incurred, as shown in Table 28. Because Group 2 requires only four machines (machines 4, 7, 8, and 9) in Machine Cell 1, the value (32.1 time units) is obtained by 4*RAND(6,10).

Tables 29-40 are minor setup-time tables for machines in machine cells. The minor setup time is obtained by the function RAND(0.5, 2.5); that is, a value between 0.5 and 2.5 is

Table 16. The setup-time table for machine 1 in Job Shop Configuration

Job	1	2
0	9	6
1	0	6
2	10	0

Table 17. The setup-time table for machine 2 in Job Shop Configuration

Job	2	5	7
0	8	9	8
2	0	10	7
5	8	0	9
7	10	7	0

Table 18. The setup-time table for machine 3 in Job Shop Configuration

Job	5	6	7	8
0	10	10	6	9
5	0	7	8	7
6	9	0	6	10
7	6	7	0	10
8	9	9	9	0

Table 19. The setup-time table for machine 4 in Job Shop Configuration

Job	1	2	3	4	5	8
0	6	6	7	10	10	9
1	0	9	8	7	6	7
2	6	0	8	9	9	8
3	9	7	0	10	8	7
4	9	9	10	0	8	10
5	8	7	10	10	0	6
8	9	8	10	6	9	0

Table 20. The setup-time table for machine 5 in Job Shop Configuration

Job	5	6	7	8
0	7	7	9	10
5	0	9	7	7
6	9	0	9	8
7	9	8	0	7
8	9	7	6	0

Table 21. The setup-time table for machine 6 in Job Shop Configuration

Job	5	7
0	7	10
5	0	10
7	6	0

Table 22. The setup-time table for machine 7 in Job Shop Configuration

Job	1	3	4	6	8	9	11
0	9	8	7	6	8	8	7
1	0	6	10	9	7	8	10
3	6	0	9	6	6	6	6
4	10	7	0	7	9	8	8
6	9	6	8	0	6	6	8
8	6	10	8	8	0	10	6
9	10	8	8	8	8	0	6
11	10	9	7	7	9	7	0

Table 23. The setup-time table for machine 8 in Job Shop Configuration

Job	1	2	4	5	6	8
0	7	9	10	6	6	7
1	0	6	10	8	7	6
2	7	0	8	6	8	7
4	6	10	0	7	7	10
5	9	6	6	0	10	7
6	6	8	7	9	0	10
8	10	6	6	6	10	0

Table 24. The setup-time table for machine 9 in Job Shop Configuration

Job	1	3	5
0	10	6	10
1	0	8	9
3	9	0	6
5	7	7	0

Table 25. The setup-time table for machine 10 in Job Shop Configuration

Job	11
0	9
11	0

Table 26. The setup-time table for machine 11 in Job Shop Configuration

Job	9	10	11
0	9	9	6
9	0	6	7
10	9	0	10
11	9	8	0

Table 27. The setup-time table for machine 12 in Job Shop Configuration

Job	9	10	11
0	7	6	10
9	0	7	8
10	9	0	7
11	9	9	0

Table 28. The major setup-time table for machine cells in traditional cellular configuration.

	Machine Cell 1			Machine Cell 2			Machine Cell 3		
Group	1	2	3	1	2	3	1	2	3
0	49.2	38.6	9.2	8.1	30.5				23.8
1	0.0	32.1	8.2	0.0	26.3				
2	48.0	0.0	9.9	7.5	0.0				
3	34.9	25.3	0.0						0.0

Table 29. The minor setup-time table for machine 1 in Machine Cell 1

Job	1	2	
0	0.7	1.0	G1
1	0.0	2.1	
2	1.9	0.0	

Table 30. The minor setup-time table for machine 4 in Machine Cell 1

Job	1	2	3	4	5	8	
0	1.7	2.1	2.1	2.1			G1
1	0.0	1.2	2.2	2.4			
2	2.0	0.0	1.4	1.6			
3	2.2	2.2	0.0	0.7			
4	0.5	0.6	1.7	0.0			
0					1.0	1.8	G2
5					0.0	1.8	
8					2.2	0.0	

Table 31. The minor setup-time table for machine 7 in Machine Cell 1

Job	1	3	4	6	8	9	11	
0	2.1	0.7	1.5					G1
1	0.0	2.4	1.5					
3	0.9	0.0	2.0					
4	0.6	1.9	0.0					
0				1.5	0.5			G2
6				0.0	2.1			
8				1.5	0.0			
0						0.6	1.7	G3
9						0.0	0.9	
11						0.5	0.0	

Table 32. The minor setup-time table for machine 8 in Machine Cell 1

Job	1	2	4	5	6	8	
0	1.1	2.2	1.4				G1
1	0.0	1.9	1.9				
2	2.0	0.0	1.9				
4	2.2	2.2	0.0				
0				2.1	1.1	1.3	G2
5				0.0	2.3	2.1	
6				1.2	0.0	0.6	
8				1.6	0.6	0.0	

Table 33. The minor setup-time table for machine 9 in Machine Cell 1

Job	1	3	5	
0	1.7	1.3		G1
1	0.0	1.9		
3	1.1	0.0		
0			0.5	G2
5			0.0	

Table 34. The minor setup-time table for machine 3 in Machine Cell 2

Job	5	6	7	8	
0	2.2	0.8	2.1	0.5	G2
5	0.0	2.1	1.9	1.9	
6	1.1	0.0	2.5	0.7	
7	1.2	1.0	0.0	1.9	
8	2.1	1.7	1.6	0.0	

Table 35. The minor setup-time table for machine 5 in Machine Cell 2

Job	5	6	7	8	
0	1.0	2.2	2.5	1.9	G2
5	0.0	2.4	0.7	1.8	
6	2.5	0.0	2.2	1.8	
7	2.0	2.1	0.0	1.4	
8	0.8	2.0	2.2	0.0	

Table 36. The minor setup-time table for machine 2 in Machine Cell 2

Job	2	5	7	
0	2.2			G1
2	0.0			
0		1.7	2.1	G2
5		0.0	2.4	
7		0.8	0.0	

Table 37. The minor setup-time table for machine 6 in Machine Cell 2

Job	5	7	
0	1.1	1.9	G2
5	0.0	2.2	
7	1.6	0.0	

Table 38. The minor setup-time table for machine 10 in Machine Cell 3

Job	11	
0	1.7	G3
11	0.0	

Table 39. The minor setup-time table for machine 11 in Machine Cell 3

Job	9	10	11	
0	2.4	1.7	1.1	G3
9	0.0	0.5	2.0	
10	1.4	0.0	1.6	
11	1.6	0.7	0.0	

Table 40. The minor setup-time table for machine 12 in Machine Cell 3

Job	9	10	11	
0	0.8	2.3	1.8	G3
9	0.0	2.3	2.1	
10	1.1	0.0	1.0	
11	0.8	2.3	0.0	

Table 41. The major setup-time table for virtual cells in virtual cellular configuration

	Virtual Cell 1			Virtual Cell 2			Virtual Cell 3			Virtual Cell 4			Virtual Cell 5		
Groups	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3
0	35.3	25.2				24.5	13.4				19.7			19.6	
1	0	32.8					0								
2	38.7	0									0			0	
3						0									

Table 42. The minor setup-time table for machine 4 in Virtual Cell 1

Job	1	2	3	4	5	8	
0	0.5	0.9	1.5	0.8			G1
1	0	0.8	0.7	2			
2	0.8	0	1	2.3			
3	1.5	2.1	0	2.5			
4	1.6	2.1	0.5	0			
0					1.1	2.4	G2
5					0	2.4	
8					1	0	

Table 43. The minor setup-time table for machine 7 in Virtual Cell 1

Job	1	3	4	6	8	
0	0.9	1.6	0.8			G1
1	0	1.7	0.6			
3	1.9	0	1.3			
4	2.1	2.1	0			
0				1.9	1.8	G2
6				0	0.7	
8				2.2	0	

Table 44. The minor setup-time table for machine 8 in Virtual Cell 1

Job	1	2	4	5	6	8	
0	2.3	1.9	2.5				G1
1	0.0	1.1	0.9				
2	1.9	0.0	1.7				
4	0.6	1.7	0.0				
0				0.6	2.2	2.0	G2
5				0.0	0.7	0.7	
6				1.9	0.0	2.4	
8				2.4	1.4	0.0	

Table 45. The minor setup-time table for machine 9 in Virtual Cell 1

Job	1	3	5	
0	0.5	1.5		G1
1	0	0.7		
3	1.5	0		
0			1.1	G2
5			0	

Table 46. The minor setup-time table for machine 7 in Virtual Cell 2

Job	9	10	11	
0	1.3	1.7	1.1	G3
9	0.0	1.8	1.0	
10	0.8	0.0	2.5	
11	1.8	1.8	0.0	

Table 47. The minor setup-time table for machine 10 in Virtual Cell 2

Job	9	11	
0	1.3	1.1	G3
9	0.0	1.6	
11	1.1	0.0	

Table 48. The minor setup-time table for machine 11 in Virtual Cell 2

Job	9	10	11	
0	2.2	1.3	1.9	G3
9	0.0	1.1	1.2	
10	2.3	0.0	0.8	
11	0.7	2.4	0.0	

Table 49. The minor setup-time table for machine 12 in Virtual Cell 2

Job	11	
0	1.9	G3
11	0.0	

Table 50. The minor setup-time table for machine 1 in Virtual Cell 3

Job	1	2	
0	0.6	2.5	G1
1	0.0	1.8	
2	2.1	0.0	

Table 51. The minor setup-time table for machine 2 in Virtual Cell 3

Job	1	2	
0	0.7	0.6	G1
1	0.0	0.8	
2	1.8	0.0	

Table 52. The minor setup-time table for machine 3 in Virtual Cell 4

Job	5	6	7	8	
0	1.1	2.0	2.2	1.4	G2
5	0.0	1.2	1.5	2.1	
6	2.1	0.0	2.1	1.0	
7	0.8	1.2	0.0	0.5	
8	0.9	0.9	1.9	0.0	

Table 53. The minor setup-time table for machine 5 in Virtual Cell 4

Job	5	6	7	8	
0	1.5	1.9	1.4	2.2	G2
5	0.0	1.7	1.9	2.2	
6	1.1	0.0	2.1	1.4	
7	1.6	1.5	0.0	2.4	
8	1.1	0.7	0.9	0.0	

Table 54. The minor setup-time table for machine 2 in Virtual Cell 5

Job	5	7	
0	1.3	2.3	
5	0.0	0.7	G2
7	0.5	0.0	

Table 55. The minor setup-time table for machine 6 in Virtual Cell 5

Job	5	7	
0	1.2	1.3	
5	0.0	1.4	G2
7	1.2	0.0	

randomly generated by the function. For instance, the minor setup-time table for machine 4 in Machine Cell 1 is shown as Table 30. When machine 4 progresses from Job 5 to Job 8 (both of which belong to Group 2), the required minor setup time is 1.8 time units.

Tables 41-55 are the required setup-time tables for virtual cellular configuration. As shown in Figure 52, there are five virtual cells for Production Session 1 in Example 1. Table 41 is the major setup-time table for the five virtual cells. Tables 42-55 are the minor setup-time tables for machines in virtual cells. The functions used to obtain the values of setup time in traditional cellular configuration are applied here also.

With these setup-time tables, the next task is to calculate the incurred setup time for each production configuration in a production session. In this research, the nearest neighbor method is employed to sequence part families/groups on a machine cell and jobs on a machine. The key idea of the nearest neighbor method is always to choose the lowest available value. Tie situations are broken by selecting one of the lowest values arbitrarily. Job sequences, which are different in job shop configuration, traditional cellular configuration, and virtual cellular configuration, are described in Sections 6.3.2 and 6.3.3.

6.3.2 Job Sequence in Job Shop Configuration

For the job shop configuration, the incurred setup time is the sum of all required machine setup times. The job sequences for machines are shown in Tables 56-67. For instance, Table 18 represents the setup-time table for machine 3, and four jobs (Jobs 5, 6, 7, and 8) need to visit machine 3. According to the nearest neighbor method, the job sequence on machine 3 is Job 7 → Job 5 → Job 6 → Job 8; the accumulated setup time on machine 3 is then 29 time

Table 56. The sequence on machine 1 in Job Shop Configuration

Job	Start	Setup	SubTotal
2	0	6	6
1	6	10	16

Table 57. The sequence on machine 2 in Job Shop Configuration

Job	Start	Setup	SubTotal
7	0	8	8
5	8	7	15
2	15	8	23

Table 58. The sequence on machine 3 in Job Shop Configuration

Job	Start	Setup	SubTotal
7	0	6	6
5	6	6	12
6	12	7	19
8	19	10	29

Table 59. The sequence on machine 4 in Job Shop Configuration

Job	Start	Setup	SubTotal
1	0	6	6
5	6	6	12
8	12	6	18
4	18	6	24
2	24	9	33
3	33	8	41

Table 60. The sequence on machine 5 in Job Shop Configuration

Job	Start	Setup	SubTotal
5	0	7	7
7	7	7	14
8	14	7	21
6	21	7	28

Table 61. The sequence on machine 6 in Job Shop Configuration

Job	Start	Setup	SubTotal
5	0	7	7
7	7	10	17

Table 62. The sequence on machine 7 in Job Shop Configuration

Job	Start	Setup	SubTotal
6	0	6	6
8	6	6	12
1	12	6	18
3	18	6	24
9	24	6	30
11	30	6	36
4	36	7	43

Table 63. The sequence on machine 8 in Job Shop Configuration

Job	Start	Setup	SubTotal
5	0	6	6
2	6	6	12
1	12	7	19
8	19	6	25
4	25	6	31
6	31	7	38

Table 64. The sequence on machine 9 in Job Shop Configuration

Job	Start	Setup	SubTotal
3	0	6	6
5	6	6	12
1	12	7	19

Table 65. The sequence on machine 10 in Job Shop Configuration

Job	Start	Setup	SubTotal
11	0	9	9

Table 66. The sequence on machine 11 in Job Shop Configuration

Job	Start	Setup	SubTotal
11	0	6	6
10	6	8	14
9	14	9	23

Table 67. The sequence on machine 12 in Job Shop Configuration

Job	Start	Setup	SubTotal
10	0	6	6
11	6	7	13
9	13	9	22

units, as shown in Table 58. The accumulated setup times on all machines are added together, and the incurred setup time of the production session in job shop configuration is thereby seen to be 308 time units.

6.3.3 Job Sequence in Traditional and Virtual Cellular Configurations

For traditional and virtual cellular configurations, the nearest neighbor method is applied, and the procedure for computing the incurred setup time is as follows:

Step 1: Sequence part families/groups on each machine/virtual cell. The nearest neighbor method is employed to sequence part families/groups according to the associated major setup times.

Step 2: Sequence jobs in the same part family/group on each required machine in a machine/ virtual cell. The nearest neighbor method is employed to sequence jobs in the same part family/group according to the associated minor setup times.

Step 3: Accumulate all major setup times and minor setup times in a machine/virtual cell.

Step 4: Sum the setup times of all machine/virtual cells; the value obtained is the incurred setup time for the production session.

By using the procedure just described, the sequences on cells and machines in traditional cellular configuration and virtual cellular configuration are obtained, as shown in Tables 68 - 75. For example, in the traditional cellular configuration, Machine Cell 2 is used to produce Group 1 and Group 2, as shown in Table 28. Using the procedure described, the sequence of the two part groups in Machine Cell 2 is Group 1 → Group 2, as shown in Table 69.

The next step is to sequence the jobs in a group on required machines in a cell. For example, the Group 2 jobs that need to visit machine 3 in Machine Cell 2 are Jobs 5, 6, 7, and 8. The minor setup-time table for these four jobs on machine 3 is presented in Table 34. According to the nearest neighbor method, the job sequence on machine 3 is Job 8 → Job 7 → Job 6 → Job 5. After the part families/groups and jobs are arranged on Machine Cell 2, the major and minor setup times are accumulated, and the setup time on Machine Cell 2 is seen to be 53.3 time units, as shown in Table 69.

The last step is to sum the setup times of all machine cells; the incurred setup time of Production Session 1 in Example 1 is thereby seen to be 198.6 (111.3 + 53.3 + 34) time units, under traditional cellular configuration.

The same procedure is applied to virtual cellular configuration. The related sequences in virtual cells and machines are presented in Tables 71-75. Then, the incurred setup time of Production Session 1 in Example 1 is 201.3 (91.8 + 36 + 18.2 + 31.2 + 24.1) time units, under virtual cellular configuration.

6.3.4 Comparison Results of Setup Time

The incurred setup times are summarized in Table 76, in which the lowest setup time of a production session is bordered. For instance, the lowest setup time of Production Session 8 in Example 1 under virtual cellular configuration is 183.8 time units, which is highlighted. The ten incurred setup times of each example are accumulated to give the subtotal setup time, denoted "SubTotal" in Table 76. In addition, all incurred setup times are summed to give the total setup time (denoted "Total") at the bottom of the table. For instance, the subtotal setup time of Example 2 under traditional cellular configuration is 1641 time units, and the total setup time under job shop configuration is 12235 time units.

Table 68. The sequence on Machine Cell 1 in Traditional Cellular Configuration

Group	Major	Machine	Job	Minor
3	9.2	7	9	0.6
			11	0.9
2	25.3	4	5	1.0
			8	1.8
		7	8	0.5
			6	1.5
		8	6	1.1
			8	0.6
			5	1.6
		9	5	0.5
1	48.0	1	1	0.7
			2	2.1
		4	1	1.7
			2	1.2
			3	1.4
			4	0.7
		7	3	0.7
			1	0.9
			4	2.0
		8	1	1.1
			2	1.9
			4	1.9
		9	3	1.3
			1	1.1
		SubTotal		111.3

Table 69. The sequence on Machine Cell 2 in Traditional Cellular Configuration

Group	Major	Machine	Job	Minor
1	8.1	2	2	2.2
2	26.3	3	8	0.5
			7	1.6
			6	1.0
			5	1.1
		5	5	1.0
			7	0.7
			8	1.4
			6	2.0
		2	5	1.7
			7	2.4
		6	5	1.1
			7	2.2
		SubTotal		53.3

Table 70. The sequence on Machine Cell 3 in Traditional Cellular Configuration

Group	Major	Machine	Job	Minor
3	23.8	10	11	1.7
		11	11	1.1
			10	0.7
			9	1.4
		12	9	0.8
			11	2.1
			10	2.3
		SubTotal		34.0

Table 71. The sequence on Virtual Cell 1 in Virtual Cellular Configuration

Group	Major	Machine	Job	Minor
2	25.2	4	5	1.1
			8	2.4
		7	8	1.8
			6	2.2
		8	5	0.6
			6	0.7
			8	2.4
		9	5	1.1
1	38.7	4	1	0.5
			3	0.7
			2	2.1
			4	2.3
		7	4	0.8
			1	2.1
			3	1.7
		8	2	1.9
			4	1.7
			1	0.6
		9	1	0.5
			3	0.7
		SubTotal		91.8

Table 72. The sequence on Virtual Cell 2 in Virtual Cellular Configuration

Group	Major	Machine	Job	Minor
3	24.5	11	11	1.1
			9	1.8
			10	1.8
		7	11	1.1
			9	1.1
		12	10	1.3
			11	0.8
			9	0.7
		10	11	1.9
			SubTotal	36.0

Table 73. The sequence on Virtual Cell 3 in Virtual Cellular Configuration

Group	Major	Machine	Job	Minor
1	13.4	1	1	0.6
			2	1.8
		2	2	0.6
			1	1.8
			SubTotal	18.2

Table 74. The sequence on Virtual Cell 4 in Virtual Cellular Configuration

Group	Major	Machine	Job	Minor
2	19.7	3	5	1.1
			6	1.2
			8	1.0
			7	1.9
		5	7	1.4
			6	1.5
			5	1.1
			8	2.2
			SubTotal	31.2

Table 75. The sequence on Virtual Cell 5 in Virtual Cellular Configuration

Group	Major	Machine	Job	Minor
2	19.6	2	5	1.3
			7	0.7
		6	5	1.2
			7	1.4
			SubTotal	24.1

Table 76. The setup-time comparison table

		Job Shop Configuration (JS)	Traditional Cellular Configuration (MC)	Virtual Cellular Configuration (VC)
Example 1	Production Session 1	308.0	198.6	201.3
	Production Session 2	331.0	200.3	231.6
	Production Session 3	302.0	139.2	158.9
	Production Session 4	362.0	157.1	180.9
	Production Session 5	365.0	223.9	239.2
	Production Session 6	205.0	142.0	144.0
	Production Session 7	276.0	130.7	167.9
	Production Session 8	302.0	192.4	183.8
	Production Session 9	441.0	231.4	308.2
	Production Session 10	448.0	321.8	300.5
	SubTotal	3340.0	1937.4	2116.2
Example 2	Production Session 1	206.0	160.7	150.1
	Production Session 2	241.0	187.0	251.3
	Production Session 3	162.0	156.6	184.3
	Production Session 4	286.0	199.2	289.0
	Production Session 5	179.0	129.3	146.8
	Production Session 6	124.0	162.0	160.1
	Production Session 7	357.0	218.1	312.9
	Production Session 8	179.0	107.6	95.1
	Production Session 9	215.0	137.9	174.4
	Production Session 10	328.0	192.5	298.9
	SubTotal	2277.0	1641.8	2062.8
Example 3	Production Session 1	154.0	108.8	131.2
	Production Session 2	150.0	113.2	128.4
	Production Session 3	256.0	110.4	182.3
	Production Session 4	136.0	86.1	99.7
	Production Session 5	328.0	114.5	177.4
	Production Session 6	202.0	123.7	104.6
	Production Session 7	228.0	157.6	212.9
	Production Session 8	246.0	140.1	179.6
	Production Session 9	201.0	101.9	142.6
	Production Session 10	302.0	145.4	174.6
	SubTotal	2203.0	1201.7	1533.2
Example 4	Production Session 1	207.0	161.8	210.3
	Production Session 2	221.0	185.8	230.1
	Production Session 3	228.0	193.0	221.9
	Production Session 4	258.0	209.0	246.5
	Production Session 5	199.0	124.0	138.7
	Production Session 6	258.0	204.0	279.3
	Production Session 7	200.0	133.0	155.2
	Production Session 8	414.0	281.8	370.6
	Production Session 9	223.0	160.4	200.5
	Production Session 10	196.0	124.9	142.2
	SubTotal	2404.0	1777.5	2195.2
Example 5	Production Session 1	179.0	136.5	142.6
	Production Session 2	250.0	173.4	209.1
	Production Session 3	197.0	163.8	178.9
	Production Session 4	189.0	134.5	162.0
	Production Session 5	204.0	141.0	154.7
	Production Session 6	245.0	166.5	197.7
	Production Session 7	206.0	161.3	186.8
	Production Session 8	118.0	109.3	106.9
	Production Session 9	157.0	107.7	148.3
	Production Session 10	266.0	231.2	228.8
	SubTotal	2011.0	1525.2	1715.8
	TOTAL	12235.0	8083.6	9623.2
	%	100%	66.07%	78.65%

Furthermore, total setup times as percentages are compared with each other. If the percentage in job shop configuration is 100%, then the percentages are 66.07% and 78.65% under traditional cellular configuration and virtual cellular configuration, respectively (Table 76). As expected, traditional and virtual cellular configurations perform better in terms of setup time than job shop configuration. As described in the literature, the major or common setup used in a part family/group saves a great deal of setup time in the experiment.

Virtual cellular configurations incurred higher total setup time than traditional cellular configurations because of the number of generated cells and the sharing concept. Given the same set of product mix, virtual cells are extracted from the job routings naturally; however, machine cells are formed by specifying the desired number of machine cells. On average, virtual cells are smaller than traditional machine cells, and the number of virtual cells is larger than the number of traditional machine cells. Therefore, in general, a part family/group of jobs will visit more virtual cells than traditional machine cells. With each visit of a part family/group, a cell will incur a major setup. Because many more major setups are incurred under virtual cellular configuration than under traditional cellular configuration, the setup time required is higher for the virtual cellular configuration.

The sharing concept is another reason for the higher total setup time for virtual cellular configuration. For example, suppose in traditional cellular configuration, machine A serves only machine cell 3. Parts that need machine A in the same part family would visit cell 3; hence, the number of major setups would be incurred only once. However, in virtual cellular configuration, machine A could be shared by multiple cells (for example, cells 3 and 5). Parts that need machine A in the same part family might visit either cell 3 or cell 5 to have their needs satisfied. Hence, the number of major setups would be doubled (one major setup for cell 3 and one for cell 5). Since the number of major setups is greater for virtual cellular configuration than for traditional cellular configuration, virtual cellular configuration requires higher total setup time.

Based on the information in Table 76, the following observations can be made:

(1) Traditional cellular configuration has the lowest setup time. By grouping jobs into part families/groups, the incurred setup time is reduced significantly.

(2) Virtual cellular configuration can also enjoy reduced setup time by grouping jobs into part families/groups. Although the incurred setup time is higher under virtual cellular configuration than under traditional cellular configuration, virtual cellular configuration has a unique advantage: Unlike traditional machine cells, virtual cells are formed naturally and no artificial parameter needs to be specified.

(3) The product mix in a production session might need or fit a particular production configuration. If the incurred setup time is used to determine which production configuration a shop should use, then a shop might choose job shop configuration (124 time units), traditional cellular configuration (218.1 time units), and virtual cellular configuration (95.1 time units) for Production Session 6, 7, and 8, respectively, in Example 2. That is, a shop could adapt its production configuration to a particular production session.

(4) According to (3), the idea of virtual production system has emerged and is proposed as a means to switch the production configuration of a shop from one production session to another in order to enjoy the lowest possible setup time.

6.4 Comparison of Material Handling Distance

The second measure used in the study is the total material handling distance. Machines are assumed to be unmovable, and AGVs perform the material handling task in a shop. The associated AGV guidepath network and incurred material handling distance are the major concerns in the comparison.

As described in Section 1.2.2 and shown in Figure 6, traditional flow network and virtual flow network are the two network considerations in the networking module of virtual production system. A traditional flow network is a fixed AGV guidepath network that once given cannot be changed. The associated traditional flow networks for the five examples are shown in Figures 47-51.

In contrast, a virtual flow network is a virtual AGV guidepath network that can be redesigned as the product mix changes. The AGV guidepath network design procedure in Chapter 4 is employed, and then the associated AGV guidepath network changes as the product mix changes.

Computing the material handling distance requires two files, a distance matrix file and a flow volume file. The first file describes the distance relationship between nodes in a layout, as shown in Table 8. One could easily convert a layout into such a distance matrix file. The other file represents the flow volumes between machines, as shown in Table 9. A flow volume file could be transferred from the product mix (in Appendix D) of a production session in examples. The associated distance matrix files and flow volume files are available upon request from the author. By using the two files, the incurred material handling distance of a production session can be computed. For instance, the incurred material handling distances for Production Session 1 in Example 2 are 647,590 and 413,370 distance units in traditional flow network and virtual flow network, respectively, as shown in Table 77. Note that the traditional flow network and the virtual flow network in Production Session 1 of Example 1 is designed by using the proposed procedure (see Chapter 4); thus the material handling distances are the same (78,150 distance units).

The material handling distances of all production sessions are added together to obtain the total material handling distance, denoted as "Total" in Table 77. The total material handling distances are 26,788,322 and 18,571,369 distance units in traditional flow network and virtual flow network, respectively. In addition, if the percentage of traditional flow network is assumed to be 100%, then the percentage is 69.32% for virtual flow network, as shown in the bottom of Table 77. Obviously, virtual flow network outperforms traditional flow network in almost all production sessions. The information in Table 77 suggests that the associated AGV guidepath network should be updated as the product mix changes, in order to minimize the material handling distance.

6.5 Comparison of Weighted Performance Value

The third measure employed in the study is the weighted performance value. The weighted performance value is calculated by combining the setup time and the material handling distance. To obtain such a performance value, the units employed in the two measurements must be unified. In this study, the factor DT (Distance to Time) is

Table 77. The comparison table of material handling distance.

		Traditional Flow Network	Virtual Flow Network
Example 1	Production Session 1	78150	78150
	Production Session 2	87300	84000
	Production Session 3	100800	84800
	Production Session 4	185300	175400
	Production Session 5	169750	160100
	Production Session 6	103800	90000
	Production Session 7	91050	79950
	Production Session 8	151300	126500
	Production Session 9	394550	348750
	Production Session 10	332500	300950
	SubTotal	1694500	1528600
Example 2	Production Session 1	647590	413370
	Production Session 2	621410	491070
	Production Session 3	368840	368460
	Production Session 4	1162060	864420
	Production Session 5	261110	252410
	Production Session 6	112090	96210
	Production Session 7	843480	591100
	Production Session 8	513680	395720
	Production Session 9	649420	535340
	Production Session 10	1270120	967060
	SubTotal	6449800	4975160
Example 3	Production Session 1	158578	149497
	Production Session 2	77314	74227
	Production Session 3	198967	186052
	Production Session 4	53860	49720
	Production Session 5	215324	212757
	Production Session 6	141038	117150
	Production Session 7	326316	307284
	Production Session 8	194915	189012
	Production Session 9	129889	98018
	Production Session 10	276698	267478
	SubTotal	1772899	1651195
Example 4	Production Session 1	200725	120803
	Production Session 2	193285	113269
	Production Session 3	186317	114433
	Production Session 4	251830	187538
	Production Session 5	222394	137746
	Production Session 6	223415	166101
	Production Session 7	111028	88308
	Production Session 8	273350	202370
	Production Session 9	135666	123906
	Production Session 10	122863	78190
	SubTotal	1920873	1332664
Example 5	Production Session 1	880050	743800
	Production Session 2	1732200	919250
	Production Session 3	1519700	917950
	Production Session 4	1372250	720300
	Production Session 5	1447550	894500
	Production Session 6	2192800	1307000
	Production Session 7	1808450	1265050
	Production Session 8	1121300	601350
	Production Session 9	1157850	436350
	Production Session 10	1718100	1278200
	SubTotal	14950250	9083750
	TOTAL	**26788322**	**18571369**
	%	**100%**	**69.32%**

developed to convert "distance unit" into "time unit". Note that the value of DT might differ between shops or companies. The weighted performance value of a production system could be computed by using the following function:

Weighted Performance Value = Setup Time + Material Handling Distance * DT

The lower a weighted performance value, the better a shop's performance. To calculate a performance value, the values of setup time (Table 76) and material handling distance (Table 77) are extracted and computed for each production session. With DT = 0.1, the performance values of a production session under different production systems are shown in Tables 78. For instance, in Production Session 3 of Example 1, the following performance values of the various production systems are computed:

Job shop (JS/FN):	302.0 + 0.1 * 100800 =	10382.0
Traditional Cellular Manufacturing (MC/FN):	139.2 + 0.1 * 100800 =	10219.2
Virtual Production System Type I (VC/FN):	158.9 + 0.1 * 100800 =	10238.9
Virtual Production System Type II (JS/VN):	302.0 + 0.1 * 84800 =	8782.0
Virtual Production System Type III (MC/VN):	139.2 + 0.1 * 84800 =	8619.2
Virtual Production System Type IV (VC/VN):	158.9 + 0.1 * 84800 =	8638.9

The obtained weighted performance values are shown in Table 78. Note here, total setup time, total material handling distance, and weighted performance value, are represented as Setup, MH. Dist., and Weighted, respectively, in Table 78. In addition, the best performance among production systems in a production session is highlighted in the table.

As shown in Table 78, virtual production system Types II (JS/VN), III (MC/VN), and IV (VC/VN) outperform job shop (JS/FN), traditional cellular manufacturing (MC/FN), and virtual production system Type I (VC/FN) in almost all production sessions. For instance, in Table 77, the weighted performance values of Example 1 (in the row of subtotal) are 172790.0, 171387.4, 171566.2, 156200.0, 154797.4, and 154975.2 time units for JS/FN, MC/FN, VC/FN, JS/VN, MC/VN, and VC/VN, respectively. Virtual flow networks

Table 78. The comparison of production system types with unmovable machines

		JS/FN			MC/FN			VC/FN		
		Setup	MH. Dist.	Weighted	Setup	MH. Dist.	Weighted	Setup	MH. Dist.	Weighted
Example 1	Prod. Session 1	308.0	78150.0	8123.0	198.6	78150.0	8013.6	201.3	78150.0	8016.3
	Prod. Session 2	331.0	87300.0	9061.0	200.3	87300.0	8930.3	231.6	87300.0	8961.6
	Prod. Session 3	302.0	100800.0	10382.0	139.2	100800.0	10219.2	158.9	100800.0	10238.9
	Prod. Session 4	362.0	185300.0	18892.0	157.1	185300.0	18687.1	180.9	185300.0	18710.9
	Prod. Session 5	365.0	169750.0	17340.0	223.9	169750.0	17198.9	239.2	169750.0	17214.2
	Prod. Session 6	205.0	103800.0	10585.0	142.0	103800.0	10522.0	144.0	103800.0	10524.0
	Prod. Session 7	276.0	91050.0	9381.0	130.7	91050.0	9235.7	167.9	91050.0	9272.9
	Prod. Session 8	302.0	151300.0	15432.0	192.4	151300.0	15322.4	183.8	151300.0	15313.8
	Prod. Session 9	441.0	394550.0	39896.0	231.4	394550.0	39686.4	308.2	394550.0	39763.2
	Prod. Session 10	448.0	332500.0	33698.0	321.8	332500.0	33571.8	300.5	332500.0	33550.5
	SubTotal	3340.0	1694500.0	**172790.0**	1937.4	1694500.0	**171387.4**	2116.3	1694500.0	**171566.3**
Example 2	Prod. Session 1	206.0	647590.0	64965.0	160.7	647590.0	64919.7	150.1	647590.0	64909.1
	Prod. Session 2	241.0	621410.0	62382.0	187.0	621410.0	62328.0	251.3	621410.0	62392.3
	Prod. Session 3	162.0	368840.0	37046.0	156.6	368840.0	37040.6	184.3	368840.0	37068.3
	Prod. Session 4	286.0	1162060.0	116492.0	199.2	1162060.0	116405.2	289.0	1162060.0	116495.0
	Prod. Session 5	179.0	261110.0	26290.0	120.3	261110.0	26231.3	146.8	261110.0	26257.8
	Prod. Session 6	124.0	112090.0	11333.0	162.0	112090.0	11371.0	160.1	112090.0	11369.1
	Prod. Session 7	357.0	843480.0	84705.0	218.1	843480.0	84566.1	312.9	843480.0	84660.9
	Prod. Session 8	179.0	513680.0	51547.0	107.6	513680.0	51475.6	95.1	513680.0	51463.1
	Prod. Session 9	215.0	649420.0	65157.0	137.9	649420.0	65079.9	174.4	649420.0	65116.4
	Prod. Session 10	328.0	1270120.0	127340.0	192.5	1270120.0	127204.5	298.9	1270120.0	127310.9
	SubTotal	2277.0	6449800.0	**647257.0**	1641.9	6449800.0	**646621.9**	2062.9	6449800.0	**647042.9**
Example 3	Prod. Session 1	154.0	158578.0	16011.8	108.8	158578.0	15966.6	131.2	158578.0	15989.0
	Prod. Session 2	150.0	77314.0	7881.4	113.2	77314.0	7844.6	128.4	77314.0	7859.8
	Prod. Session 3	256.0	198967.0	20152.7	110.4	198967.0	20007.1	182.3	198967.0	20079.0
	Prod. Session 4	136.0	53860.0	5522.0	86.1	53860.0	5472.1	99.7	53860.0	5485.7
	Prod. Session 5	328.0	215324.0	21860.4	114.5	215324.0	21646.9	177.4	215324.0	21709.8
	Prod. Session 6	202.0	141038.0	14305.8	123.7	141038.0	14227.5	104.6	141038.0	14208.4
	Prod. Session 7	228.0	326316.0	32859.6	157.6	326316.0	32789.2	212.9	326316.0	32844.5
	Prod. Session 8	246.0	194915.0	19737.5	140.1	194915.0	19631.6	179.6	194915.0	19671.1
	Prod. Session 9	201.0	129889.0	13189.9	101.9	129889.0	13090.8	142.6	129889.0	13131.5
	Prod. Session 10	302.0	276698.0	27971.8	145.4	276698.0	27815.2	174.6	276698.0	27844.4
	SubTotal	2203.0	1772899.0	**179492.9**	1201.7	1772899.0	**178491.6**	1533.3	1772899.0	**178823.2**
Example 4	Prod. Session 1	207.0	200725.0	20279.5	161.8	200725.0	20234.3	210.3	200725.0	20282.8
	Prod. Session 2	221.0	193285.0	19549.5	185.8	193285.0	19514.3	230.1	193285.0	19558.6
	Prod. Session 3	228.0	186317.0	18859.7	193.0	186317.0	18824.7	221.9	186317.0	18853.6
	Prod. Session 4	258.0	251830.0	25441.0	209.0	251830.0	25392.0	246.5	251830.0	25429.5
	Prod. Session 5	199.0	222394.0	22438.4	124.0	222394.0	22363.4	138.7	222394.0	22378.1
	Prod. Session 6	258.0	223415.0	22599.5	204.0	223415.0	22545.5	279.3	223415.0	22620.8
	Prod. Session 7	200.0	111028.0	11302.8	133.0	111028.0	11235.8	155.2	111028.0	11258.0
	Prod. Session 8	414.0	273350.0	27749.0	281.8	273350.0	27616.8	370.6	273350.0	27705.6
	Prod. Session 9	223.0	135666.0	13789.6	160.4	135666.0	13727.0	200.5	135666.0	13767.1
	Prod. Session 10	196.0	122863.0	12482.3	124.9	122863.0	12411.2	142.2	122863.0	12428.5
	SubTotal	2404.0	1920873.0	**194491.3**	1777.7	1920873.0	**193865.0**	2195.3	1920873.0	**194282.6**
Example 5	Prod. Session 1	179.0	880050.0	88184.0	136.5	880050.0	88141.5	142.6	880050.0	88147.6
	Prod. Session 2	250.0	1732200.0	173470.0	173.4	1732200.0	173393.4	209.1	1732200.0	173429.1
	Prod. Session 3	197.0	1519700.0	152167.0	163.8	1519700.0	152133.8	178.9	1519700.0	152148.9
	Prod. Session 4	189.0	1372250.0	137414.0	134.5	1372250.0	137359.5	162.0	1372250.0	137387.0
	Prod. Session 5	204.0	1447550.0	144959.0	141.0	1447550.0	144896.0	154.7	1447550.0	144909.7
	Prod. Session 6	245.0	2192800.0	219525.0	166.5	2192800.0	219446.5	197.7	2192800.0	219477.7
	Prod. Session 7	206.0	1808450.0	181051.0	161.3	1808450.0	181006.3	186.8	1808450.0	181031.8
	Prod. Session 8	118.0	1121300.0	112248.0	109.3	1121300.0	112239.3	106.9	1121300.0	112236.9
	Prod. Session 9	157.0	1157850.0	115942.0	107.7	1157850.0	115892.7	148.3	1157850.0	115933.3
	Prod. Session 10	266.0	1718100.0	172076.0	231.2	1718100.0	172041.2	228.8	1718100.0	172038.8
	SubTotal	2011.0	14950250.0	**1497036.0**	1525.2	14950250.0	**1496550.2**	1715.8	14950250.0	**1496740.8**
	Total			2691067.2			2686916.1			2688455.8
	%			100%			99.8%			99.9%

Table 78. (continued)

		JS/VN			MC/VN			VC/VN		
		Setup	MH. Dist.	Weighted	Setup	MH. Dist.	Weighted	Setup	MH. Dist.	Weighted
Example 1	Prod. Session 1	308.0	78150	8123.0	198.6	78150	8013.6	201.3	78150	8016.3
	Prod. Session 2	331.0	84000	8731.0	200.3	84000	8600.3	231.6	84000	8631.6
	Prod. Session 3	302.0	84800	8782.0	139.2	84800	8619.2	158.9	84800	8638.9
	Prod. Session 4	362.0	175400	17902.0	157.1	175400	17697.1	180.9	175400	17720.9
	Prod. Session 5	365.0	160100	16375.0	223.9	160100	16233.9	239.2	160100	16249.2
	Prod. Session 6	205.0	90000	9205.0	142.0	90000	9142.0	144.0	90000	9144.0
	Prod. Session 7	276.0	79950	8271.0	130.7	79950	8125.7	167.9	79950	8162.9
	Prod. Session 8	302.0	126500	12952.0	192.4	126500	12842.4	183.8	126500	12833.8
	Prod. Session 9	441.0	348750	35316.0	231.4	348750	35106.4	308.2	348750	35183.2
	Prod. Session 10	448.0	300950	30543.0	321.8	300950	30416.8	300.5	300950	30395.5
	SubTotal	3340.0	1528600	**156200.0**	1937.4	1528600	**154797.4**	2116.3	1528600	**154976.3**
Example 2	Prod. Session 1	206.0	413370	41543.0	160.7	413370	41497.7	150.1	413370	41487.1
	Prod. Session 2	241.0	491070	49348.0	187.0	491070	49294.0	251.3	491070	49358.3
	Prod. Session 3	162.0	368460	37008.0	156.6	368460	37002.6	184.3	368460	37030.3
	Prod. Session 4	286.0	864420	86728.0	199.2	864420	86641.2	289.0	864420	86731.0
	Prod. Session 5	179.0	252410	25420.0	120.3	252410	25361.3	146.8	252410	25387.8
	Prod. Session 6	124.0	96210	9745.0	162.0	96210	9783.0	160.1	96210	9781.1
	Prod. Session 7	357.0	591100	59467.0	218.1	591100	59328.1	312.9	591100	59422.9
	Prod. Session 8	179.0	395720	39751.0	107.6	395720	39679.6	95.1	395720	39667.1
	Prod. Session 9	215.0	535340	53749.0	137.9	535340	53671.9	174.4	535340	53708.4
	Prod. Session 10	328.0	967060	97034.0	192.5	967060	96898.5	298.9	967060	97004.9
	SubTotal	2277.0	4975160	**499793.0**	1641.9	4975160	**499157.9**	2062.9	4975160	**499578.9**
Example 3	Prod. Session 1	154.0	149497	15103.7	108.8	149497	15058.5	131.2	149497	15080.9
	Prod. Session 2	150.0	74227	7572.7	113.2	74227	7535.9	128.4	74227	7551.1
	Prod. Session 3	256.0	186052	18861.2	110.4	186052	18715.6	182.3	186052	18787.5
	Prod. Session 4	136.0	49720	5108.0	86.1	49720	5058.1	99.7	49720	5071.7
	Prod. Session 5	328.0	212757	21603.7	114.5	212757	21390.2	177.4	212757	21453.1
	Prod. Session 6	202.0	117150	11917.0	123.7	117150	11838.7	104.6	117150	11819.6
	Prod. Session 7	228.0	307284	30956.4	157.6	307284	30886.0	212.9	307284	30941.3
	Prod. Session 8	246.0	189012	19147.2	140.1	189012	19041.3	179.6	189012	19080.8
	Prod. Session 9	201.0	98018	10002.8	101.9	98018	9903.7	142.6	98018	9944.4
	Prod. Session 10	302.0	267478	27049.8	145.4	267478	26893.2	174.6	267478	26922.4
	SubTotal	2203.0	1651195	**167322.5**	1201.7	1651195	**166321.2**	1533.3	1651195	**166652.8**
Example 4	Prod. Session 1	207.0	120803	12287.3	161.8	120803	12242.1	210.3	120803	12290.6
	Prod. Session 2	221.0	113269	11547.9	185.8	113269	11512.7	230.1	113269	11557.0
	Prod. Session 3	228.0	114433	11671.3	193.0	114433	11636.3	221.9	114433	11665.2
	Prod. Session 4	258.0	187538	19011.8	209.0	187538	18962.8	246.5	187538	19000.3
	Prod. Session 5	199.0	137746	13973.6	124.0	137746	13898.6	138.7	137746	13913.3
	Prod. Session 6	258.0	166101	16868.1	204.0	166101	16814.1	279.3	166101	16889.4
	Prod. Session 7	200.0	88308	9030.8	133.0	88308	8963.8	155.2	88308	8986.0
	Prod. Session 8	414.0	202370	20651.0	281.8	202370	20518.8	370.6	202370	20607.6
	Prod. Session 9	223.0	123906	12613.6	160.4	123906	12551.0	200.5	123906	12591.1
	Prod. Session 10	196.0	78190	8015.0	124.9	78190	7943.9	142.2	78190	7961.2
	SubTotal	2404.0	1332664	**135670.4**	1777.7	1332664	**135044.1**	2195.3	1332664	**135461.7**
Example 5	Prod. Session 1	179.0	743800	74559.0	136.5	743800	74516.5	142.6	743800	74522.6
	Prod. Session 2	250.0	919250	92175.0	173.4	919250	92098.4	209.1	919250	92134.1
	Prod. Session 3	197.0	917950	91992.0	163.8	917950	91958.8	178.9	917950	91973.9
	Prod. Session 4	189.0	720300	72219.0	134.5	720300	72164.5	162.0	720300	72192.0
	Prod. Session 5	204.0	894500	89654.0	141.0	894500	89591.0	154.7	894500	89604.7
	Prod. Session 6	245.0	1307000	130945.0	166.5	1307000	130866.5	197.7	1307000	130897.7
	Prod. Session 7	206.0	1265050	126711.0	161.3	1265050	126666.3	186.8	1265050	126691.8
	Prod. Session 8	118.0	601350	60253.0	109.3	601350	60244.3	106.9	601350	60241.9
	Prod. Session 9	157.0	436350	43792.0	107.7	436350	43742.7	148.3	436350	43783.3
	Prod. Session 10	266.0	1278200	128086.0	231.2	1278200	128051.2	228.8	1278200	128048.8
	SubTotal	2011.0	9083750	**910386.0**	1525.2	9083750	**909980.2**	1715.8	9083750	**910090.8**
	Total			1869371.9			1865220.8			1866760.5
	%			69.5%			69.3%			69.4%

constitute the main reason that virtual production system Types II (JS/VN), III (MC/VN), and IV (VC/VN) perform better than the other three production systems. Through redesigning of the associated AGV guidepath network, the material handling distances are reduced significantly.

As shown at the bottom of Table 78, all performance values of a production system are added together to obtain the total performance value denoted "Total". If the performance value of job shop is counted as 100%, then the percentages are 99.8%, 99.9%, 69.5%, 69.3%, and 69.4% for MC/FN, VC/FN, JS/VN, MC/VN, and VC/VN, respectively. Clearly, the production system types with virtual flow network outperform those with traditional flow network.

In addition, according to the results of this study, the value of material handling distance greatly exceeds the value of setup time in a production session. As shown in Table 78, the material handling distance dominates the setup time in the weighted performance value, even with DT = 0.1. This suggests that the total material handling distance/time should get more attention than the total setup time in the real world.

Furthermore, because a shop is allowed to select the best production system in a production session, the selected production systems for all production sessions in the experiment are summarized in Table 79. For instance, in Example 1, a shop selects virtual production system Type III (MC/VN) for Production Session 7 and switches to virtual production system Type IV (VC/VN) for Production Session 8. The total weighted performance value of these best production systems is 1865100.3 time units, as shown in Table 79.

Moreover, because the factor DT would influence the performance value of a production system, different values of DT are also examined in the proposal. The range of DT is changed from 0.1 to 0.2 by 0.01 increments, and the results are shown in Figures 53-57. Note that, because the performance values are close to each other, the lines that represent virtual production system Type II (JS/VN), III (MC/VN), and IV (VC/VN) overlap in the figures. Similarly, the lines that represent job shop (JS/FN), traditional cellular manufacturing (MC/FN), and virtual production system Type I (VC/FN) overlap in the figures.

Table 79. The selected virtual production system types for production sessions

Example	Production Session	Production System	Weighted
Example 1	Production Session 1	Virtual Production System Type III	8013.6
	Production Session 2	Virtual Production System Type III	8600.3
	Production Session 3	Virtual Production System Type III	8619.2
	Production Session 4	Virtual Production System Type III	17697.1
	Production Session 5	Virtual Production System Type III	16233.9
	Production Session 6	Virtual Production System Type III	9142.0
	Production Session 7	Virtual Production System Type III	8125.7
	Production Session 8	Virtual Production System Type IV	12830.9
	Production Session 9	Virtual Production System Type III	35106.4
	Production Session 10	Virtual Production System Type IV	30392.8
Example 2	Production Session 1	Virtual Production System Type IV	41487.1
	Production Session 2	Virtual Production System Type III	49294.0
	Production Session 3	Virtual Production System Type III	37002.6
	Production Session 4	Virtual Production System Type III	86641.2
	Production Session 5	Virtual Production System Type III	25361.3
	Production Session 6	Virtual Production System Type II	9745.0
	Production Session 7	Virtual Production System Type III	59328.1
	Production Session 8	Virtual Production System Type IV	39667.1
	Production Session 9	Virtual Production System Type III	53671.9
	Production Session 10	Virtual Production System Type III	96898.5
Example 3	Production Session 1	Virtual Production System Type III	15058.5
	Production Session 2	Virtual Production System Type III	7535.9
	Production Session 3	Virtual Production System Type III	18715.6
	Production Session 4	Virtual Production System Type III	5058.1
	Production Session 5	Virtual Production System Type III	21390.2
	Production Session 6	Virtual Production System Type IV	11819.6
	Production Session 7	Virtual Production System Type III	30886.0
	Production Session 8	Virtual Production System Type III	19041.3
	Production Session 9	Virtual Production System Type III	9903.7
	Production Session 10	Virtual Production System Type III	26893.2
Example 4	Production Session 1	Virtual Production System Type III	12242.1
	Production Session 2	Virtual Production System Type III	11512.7
	Production Session 3	Virtual Production System Type III	11636.3
	Production Session 4	Virtual Production System Type III	18962.8
	Production Session 5	Virtual Production System Type III	13898.6
	Production Session 6	Virtual Production System Type III	16814.1
	Production Session 7	Virtual Production System Type III	8963.8
	Production Session 8	Virtual Production System Type III	20518.8
	Production Session 9	Virtual Production System Type III	12551.0
	Production Session 10	Virtual Production System Type III	7943.9
Example 5	Production Session 1	Virtual Production System Type III	74516.5
	Production Session 2	Virtual Production System Type III	92098.4
	Production Session 3	Virtual Production System Type III	91958.8
	Production Session 4	Virtual Production System Type III	72164.5
	Production Session 5	Virtual Production System Type III	89591.0
	Production Session 6	Virtual Production System Type III	130866.5
	Production Session 7	Virtual Production System Type III	126666.3
	Production Session 8	Virtual Production System Type IV	60241.9
	Production Session 9	Virtual Production System Type III	43742.7
	Production Session 10	Virtual Production System Type IV	128048.8
Total			**1865100.3**

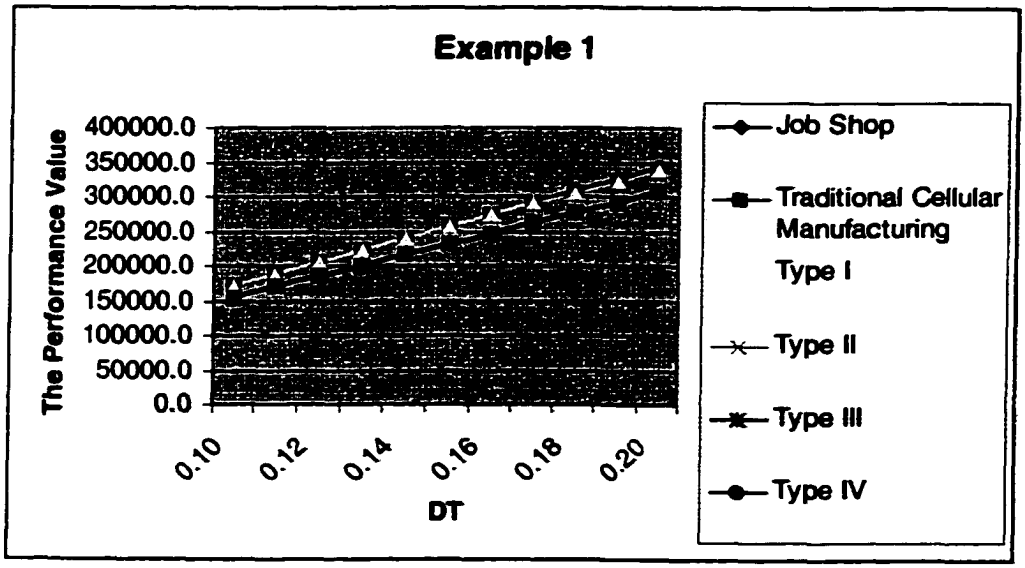


Figure 53. Performance in Example 1 with DT = 0.1~0.2

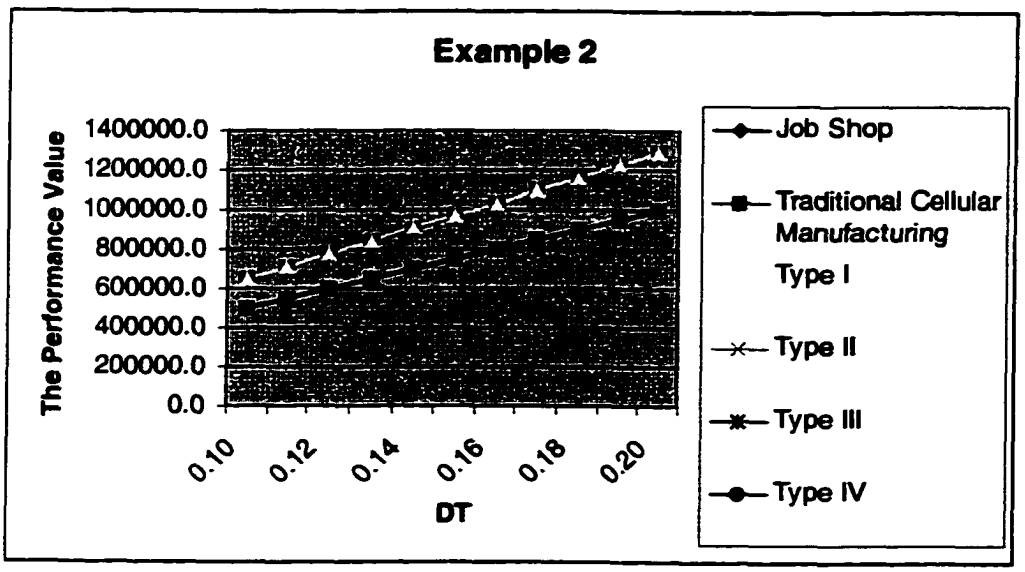


Figure 54. Performance in Example 2 with DT = 0.1~0.2

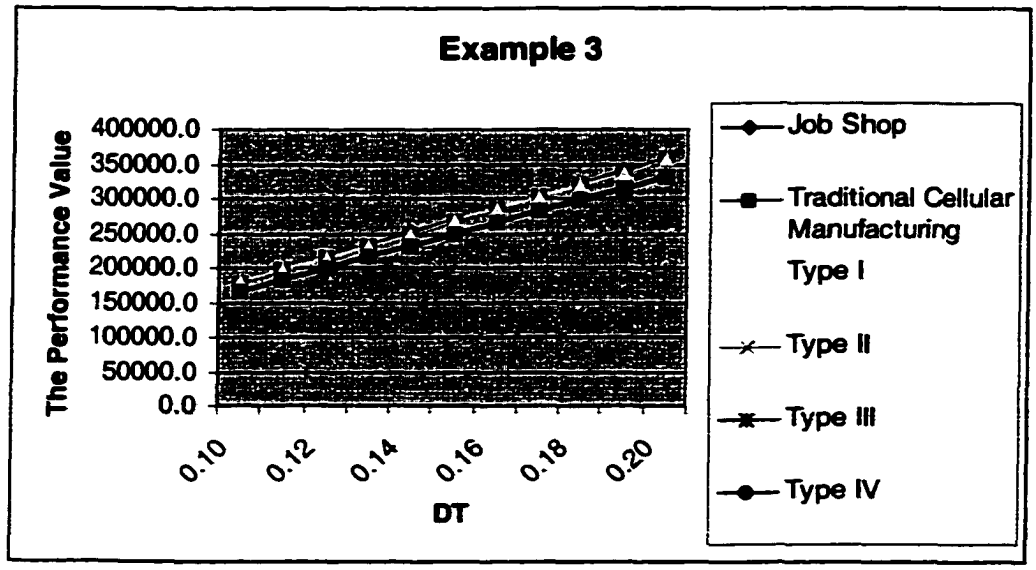


Figure 55. Performance in Example 3 with DT = 0.1~0.2

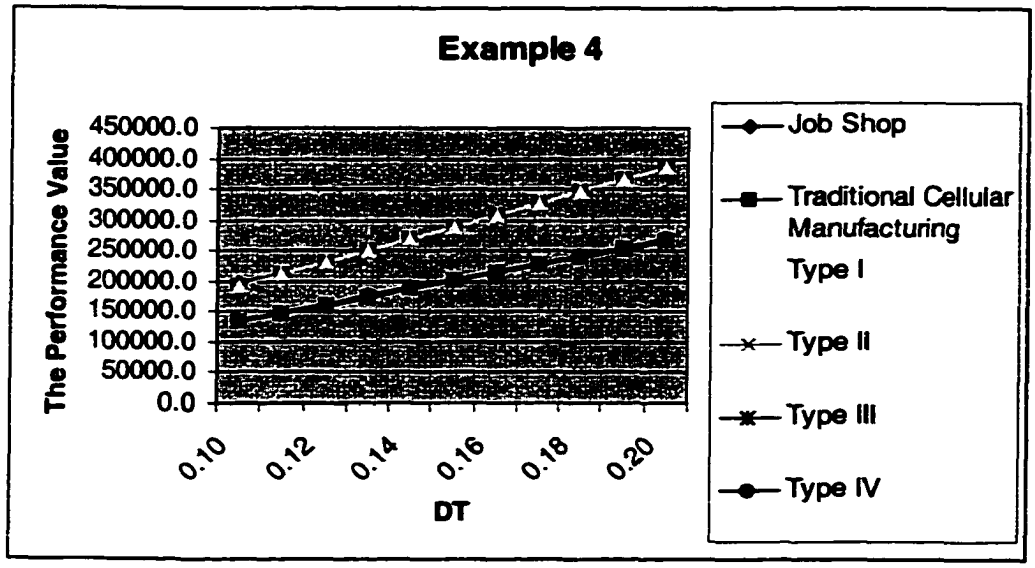


Figure 56. Performance in Example 4 with DT = 0.1~0.2

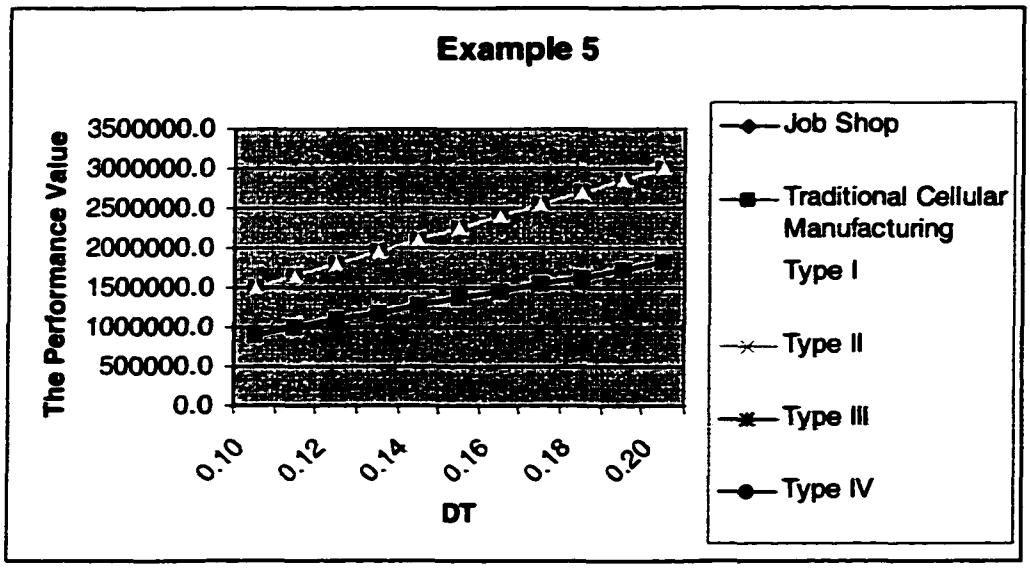


Figure 57. Performance in Example 5 with DT = 0.1~0.2

The following conclusions are based on the information obtained in this section:

(1) The production systems with virtual flow network outperform those with traditional flow network. In this research, virtual production system Types II (JS/VN), III (MC/VN), and IV (VC/VN) performed better than other production system types under the same experimental conditions. Based on the results obtained, the virtual flow network significantly improves a shop's performance.

(2) The weighted performance value of a production system is dominated by the total material handling distance. As shown in this section, the total material handling distance/time has a greater impact than the total setup time on a shop's efficiency.

(3) By employing the processing system configuration module and the networking module in virtual production system, a shop is able to select the best production system type in a production session and thus improve its performance further.

CHAPTER 7. THE IMPACT OF MOVABLE MACHINES ON A SHOP

In this chapter, the constraint of unmovable machines is relaxed; machines are no longer fixed in position, and a shop could be physically reorganized to respond to dynamically changing product mix environments. Machines with heavy flow volume between them can be located as close together as possible, so as to decrease total material handling distance. The impact of movable machines on a shop will be explored.

7.1 Introduction

One Virtual Production System characteristic described in Chapter 6 is the assumption of unmovable machines. A material handling device within a related flow network performs the material flow linkage between machines. Because machines are locked in position, one possible way to reduce total material handling distance is the use of a virtual flow network, which is redesigned whenever the product mix changes, so as to reduce the total material handling distance, as discussed in Chapter 6.

The alternative is to relax the constraint of unmovable machines; that is, to reorganize a shop physically whenever the product mix changes. With a given flow network, a shop is geographically divided into sites and to which machines are assigned on a temporary basis. The locations of machines can be reassigned as the product mix changes, so as to achieve reduction of total material handling distance.

In practice, machines in some industries, such as electronics and/or some make-to-order manufacturers, are easier to move than those in heavy industries. For the sake of efficiency, it is not unusual to reorganize a shop physically in order to respond to changes in product mix for these types of industry. When a shop is reconfigured, machines with heavy flow volume between them can be located as close together as possible, so as to increase the efficiency of total material handling distance. In this chapter, the impact of movable machines on a shop is investigated.

7.2 Assumptions

Before further examination of a shop with movable machines, several assumptions need to be stated. First, except that the constraint of unmovable machines is relaxed, the layouts and product mixes are the same as for the examples of Chapter 6. The major comparison between a shop with and without movable machines is focused on two material-handling-related measures: total material handling distance and weighted performance value.

Second, although the machines are movable, material handling is still performed by material handling equipment (such as AGVs or conveyers) within a flow network. The flow network and related loading/unloading points, once given, are fixed.

Third, a shop is geographically divided into smaller sites by its given flow network and machines are assigned to these sites on a temporary basis. When the product mix changes, the locations of machines may also change in order to minimize material handling distance.

Fourth, a shop is operated as a job shop or a cellular manufacturing shop. On the basis of these two configurations, the shop is reorganized and examined with changes in product mix. The consideration of cellular configuration is discussed next.

7.3 The Cellular Configuration Shop

In this study, two of cell formation procedures for machine grouping are compared. One is Ko's cell formation method of Chapter 3. As explained in Chapter 3, Ko's method could form machine cells very quickly without specifying any artificial parameters. The characteristics of machine cells created by Ko's method are that a machine could be shared by more than one cell and a cell could serve multiple part families. By use of the product mix data in Appendix D, the machine cells obtained with Ko's method are presented in Appendix F.

The other cell formation method is modified from Boctor's model [44]. As shown in Figure 58, the objective of the modified model is to minimize the number of exceptional parts (those parts that must visit more than one machine cell to be completed). Constraint (1) indicates that a machine could be assigned to more than one machine cell. Constraint (2) implies that a job must be assigned to only one cell, although a job might need to visit

$$\text{MIN} \quad Z = \sum_{i=1}^{M} \sum_{j=1}^{P} \frac{a_{ij}\left(\sum_{k=1}^{C} \left| x_{ik} - y_{jk} \right| \right)}{2}$$

Subject to :

$$\sum_{k=1}^{C} x_{ik} \geq 1 \qquad i = 1 \sim M \qquad (1)$$

$$\sum_{k=1}^{C} y_{jk} = 1 \qquad j = 1 \sim P \qquad (2)$$

$$\sum_{i=1}^{M} x_{ik} \leq m_1 \qquad k = 1 \sim C \qquad (3)$$

$$\sum_{i=1}^{M} x_{ik} \geq m_2 \qquad k = 1 \sim C \qquad (4)$$

$$x_{ik} \in \{0, 1\} \qquad \forall i, k \qquad (5)$$

$$y_{jk} \in \{0, 1\} \qquad \forall j, k \qquad (6)$$

Where,

- i = machine index;
- j = part index;
- k = cell index;
- M = number of machines;
- P = total number of parts;
- C = number of manufacturing cells;
- m_1 = maximum number of machines allowed in a cell;
- m_2 = minimum number of machines allowed in a cell;
- a_{ij} = volume of part j required to be processed on machine i;
- x_{ik} = binary variable indicating whether or not machine i is assigned to cell k;
- y_{jk} = binary variable indicating whether or not part j is assigned to cell k.

Figure 58. The modified version of Boctor's model

multiple cells to have its needs satisfied. Constraint (3) restricts the number of machines that a cell could contain, while Constraint (4) specifies the smallest number of machines in a cell. Constraints (5) and (6) guarantee the required integrality and non-negative properties.

To activate the model, the parameters C, m_1, and m_2 must be specified by the user. Instead of arbitrarily assigning the three parameters any value, the information of machine cells obtained by using Ko's method for each production session can be used. For example, consider Table 80, which shows five machine cells created by using Ko's method for Production Session 1 of Example 1. Machine cells 1 and 2 contain the largest number of machines, (4 machines each), while machine cells 3, 4, and 5 contain the smallest number of machines, (2 machines each). Therefore, for Production Session 1 in Example 1, C, m_1 and m_2 would be specified in the modified Boctor's model as 5, 4, and 2, respectively. Based on the information in Appendix F, the values of the three parameters vary for every production session. With the same data on product mixes as given in Appendix D, the machine cells generated by using the modified model are presented in Appendix G.

Table 80. Two Cellular Configurations for Production Session 1 of Example 1

	Ko's Method		Modified Boctor's Model	
	Machines	Cell Volume	Machines	Cell Volume
Machine Cell 1	4, 7, 8, 9	715	4, 7, 8, 9	855
Machine Cell 2	11, 7, 12, 10	190	2, 3, 5, 6	435
Machine Cell 3	1, 2	220	10, 11, 12	190
Machine Cell 4	3, 5	315	1, 10	220
Machine Cell 5	2, 6	160	10, 11, 12	0*

*: 0 indicates this is a dummy cell

The results obtained by using the modified Boctor's model for Production Session 1 in Example 1 are also presented in Table 80. Even though the information obtained by Ko's method is the same as in the modified Boctor's model, the two sets of machine cells are distinguishable. Only machine cell 1, containing machines 4, 7, 8, and 9, is produced by both cell formation methods. The two sets of cells are different from each other in their machine configurations.

As mentioned before, the constraint that a machine be assigned to only one cell is relaxed in the modified Boctor's model. However, on the basis of the cell configurations obtained in Appendix G, few machine cells take advantage of the relaxation. For example, of the twelve machines, only machine 10 is assigned to multiple cells (machine cells 3 and 4) in the modified Boctor's model for Production Session 1 in Example 1, as shown in Table 80.

Moreover, although C is specified as 5 in the modified Boctor's model for Production Session 1 of Example 1, only four machine cells are valid. As shown in Table 80, machine cells 3 and 5 have the same combination (machines 10, 11, and 12). Because there is no reason to have machine cells 3 and 5 at the same time, machine cell 5 is redundant and invalid.

Also shown in Table 80 is the cell volume for each machine cell. The cell volume is a measure of the total flow volume through the cell. For example, the cell volume for machine cell 3 created by using Ko's method is 220 units. Note here, because the two cell configurations are different, the total cell volumes might not be the same. In Table 80, the sums of cell volumes for Ko's method and the modified model are 1600 (715+190+220+315+160=1600) and 1700 (855+435+190+220=1700) units, respectively. The measure of cell volume plays an important role for machine assignment, as will be described next.

7.4 Machine Assignment to Sites on The Shop Floor

It is assumed in this study that a shop floor is partitioned into slots on which the machines are located. Because machines can be rearranged, the task is how to assign the machines to the slots so that the total material handling distance is minimized. For a job shop type organization, the machine assignment problem is very straightforward. With a fixed flow

network and flow volume between machines, the Quadratic Assignment Problem (QAP) technique is used to assign machines into sites.

As shown in Figure 59, the objective of the QAP model is to minimize the distance flow volume among machines. Constraint (1) indicates that a slot could only be occupied by one machine. Constraint (2) specifies that a machine could only be assigned to one slot. Constraint (3) guarantees the required integrality and non-negative properties.

$$MIN \quad Z = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \sum_{k=1}^{N-1} \sum_{l=k+1}^{N} x_{ik} * x_{jl} * d_{kl} * v_{ij}$$

Subject to :

$$\sum_{i=1}^{N} x_{ik} = 1 \qquad k = 1 \sim N \qquad (1)$$

$$\sum_{k=1}^{N} x_{ik} = 1 \qquad i = 1 \sim N \qquad (2)$$

$$x_{ik} \in \{0,1\} \qquad \forall i,k \qquad (3)$$

Where,

i = machine index;

j = machine index;

k = slot index;

l = slot index;

N = number of machines (slots);

x_{ik} = binary variable indicating whether or not machine i is assigned to slot k;

x_{jl} = binary variable indicating whether or not machine j is assigned to slot l;

d_{kl} = the distance between slot k and slot l;

v_{ij} = the flow volume between machine i and machine j;

Figure 59. The QAP technique

On the other hand, machine assignment is less straightforward when a shop is organized as a cellular type system, in which machines are grouped into machine cells (the cell formation is as described in Section 7.3). Instead of machines being assigned one at a time, machines belonging to the same machine cell are considered for assignment at the same time. Hence, the machine assignment problem becomes a machine cell assignment problem. A procedure developed for machine cell assignment is described below.

The machine cell location algorithm:

Step 1: Arrange cells in a non-increasing order based on their cell flow volume.

Step 2: Assign priority to each cell based on the order in Step 1. That is, the cell with the highest cell flow volume has the highest priority.

Step 3: Select the cell with the highest priority.

Step 4: Considering only the empty slots, use the Quadratic Assignment Problem (QAP) technique to assign machines in the cell into slots. Once a machine is assigned into a slot, the machine is fixed in the slot.

Step 5: Set occupied slots nonempty.

Step 6: 6.1: If all slots are occupied or no more cells exist, then go to Step 7.

6.2: Select the next highest priority cell and go to Step 4.

Step 7: Stop. Calculate the total material handling distance associated with the assignment.

Generally, the algorithm assigns machine cells into sites based on their cell flow volume for production. The cell flow volume is a measure of the total flow amount that passes through a cell. The machine cell with the largest cell flow volume has the highest priority for assignment on the layout. The QAP technique is employed to determine the best sites to assign the machines in a machine cell. The machine cell location algorithm is invoked sequentially, until all sites are occupied or no machine cells remain unassigned.

For the sake of illustration, consider machine cells obtained by using the modified Boctor's model in Table 80. According to their cell volume in Table 80, the priority order for the machine cells is cell 1 (855 units), cell 2 (435 units), cell 4 (220 units), and then cell 3

(190 units). The sequence of the priority order is employed for machine cell assignment as presented below.

First, the shop layout and the slots are as shown in Figure 60. Next, the cell with the highest priority, machine cell 1, is assigned into sites by using the QAP technique. As shown in Figure 61, machines 4, 7, 8, and 9, which belong to machine cell 1, occupy sites 1, 2, 3, and 4, respectively. The occupied sites are set nonempty. Then, the cell with the second highest priority, machine cell 2, is assigned to sites without considering any nonempty slots, as shown in Figure 62. The machine cell assignment is continued until all sites are occupied or no more cells exist. Note here, as shown in Figures 63 and 64, machine 10 provides service for machine cells 3 and 4. The task of machine/machine cell assignment is repeated from one production session to another in response to changes in the product mix.

7.5 A Shop with Movable Machines and Virtual Flow Network

To reduce the incurred total material handling distance in a changing product mix environment, two options have been discussed so far. One is the use of movable machines; the other one is the use of a virtual flow network. For a shop with movable machines, an

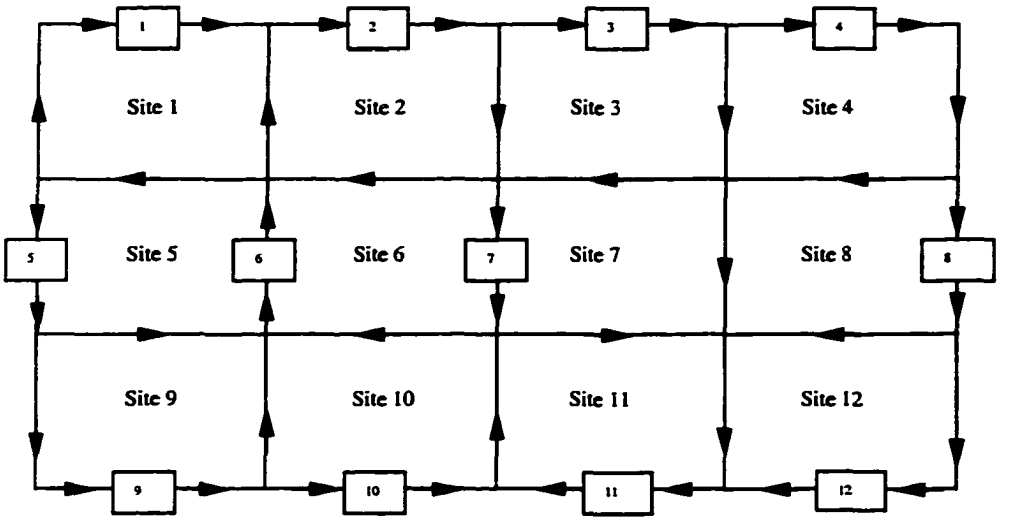


Figure 60. The shop with a given flow network of Example 1

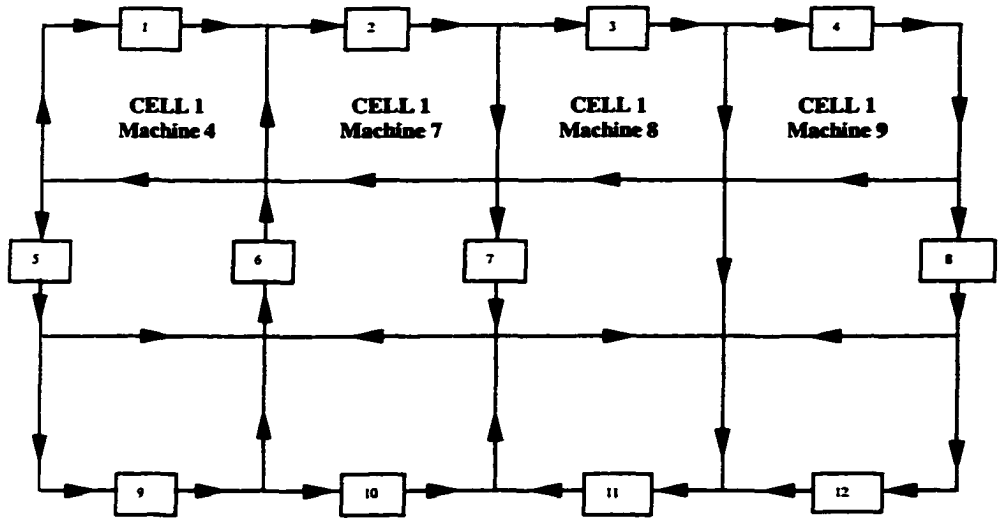


Figure 61. The location of machine cell 1

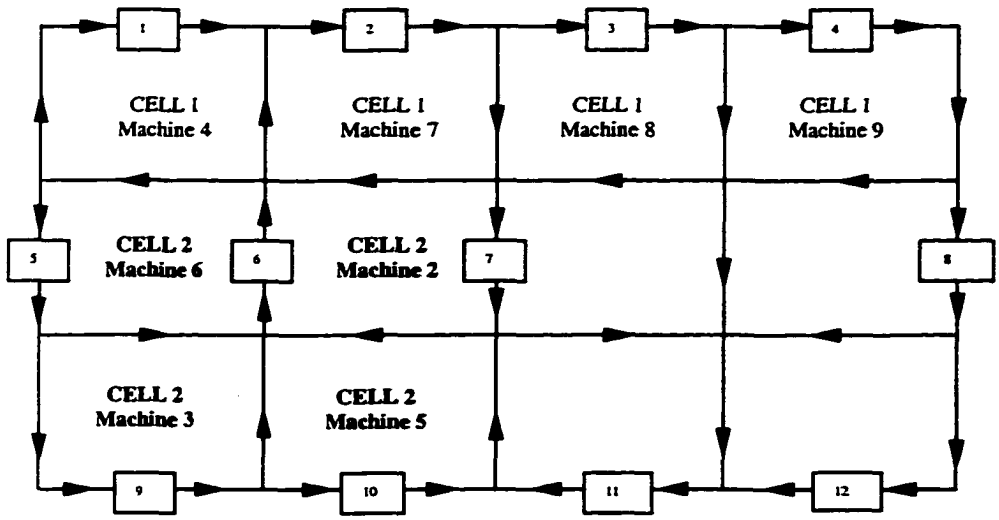


Figure 62. The location of machine cell 2

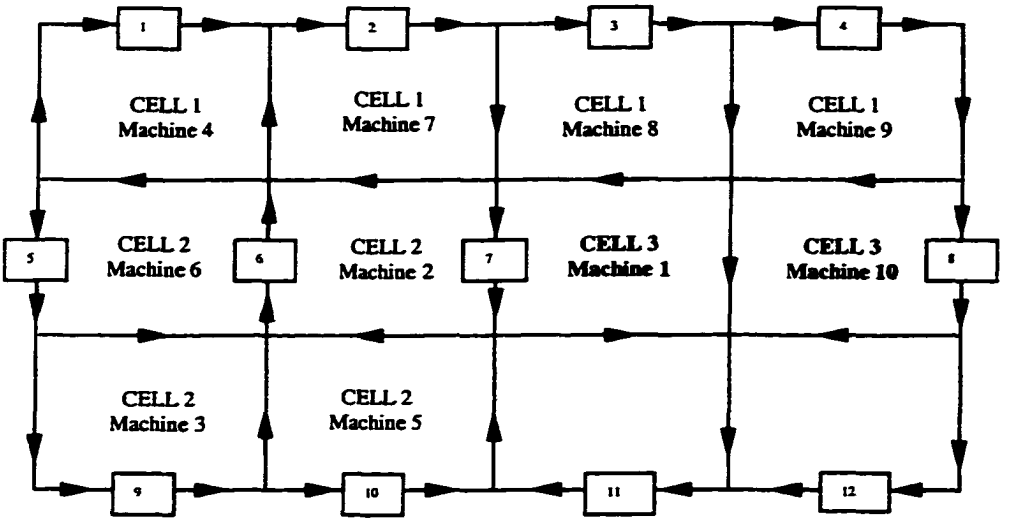


Figure 63. The location of machine cell 3

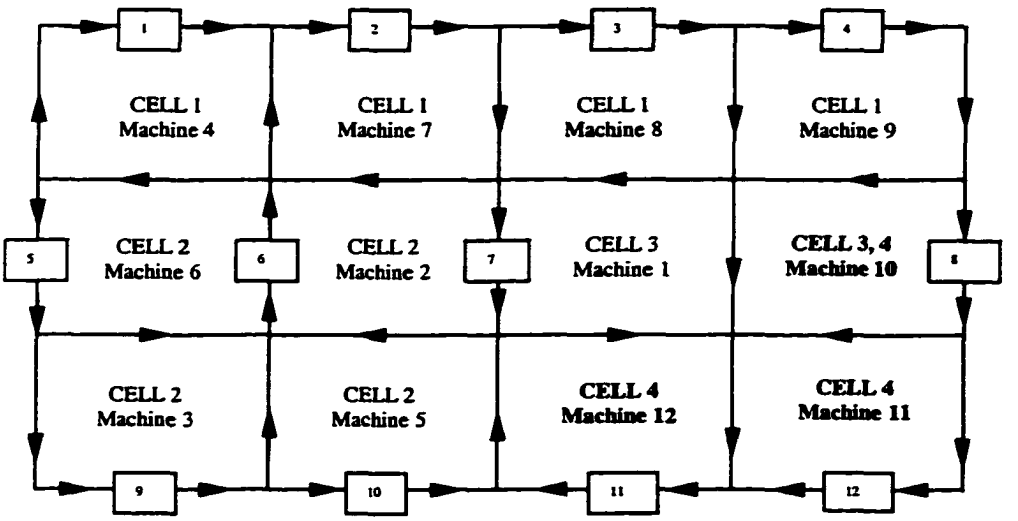


Figure 64. The location of machine cell 4

associated flow network is assumed to exist. Machines are reassigned to sites as the product mix changes, thus reducing total material handling distance.

For a shop with a virtual flow network, machines are assumed to be fixed in position. The associated flow network is redesigned in response to changes in product mixes, so that the total material handling distance may be reduced.

In addition to these two options, another option, a shop with both movable machines and a virtual flow network, is possible. That is, a shop could reorganize its machines and redesign its flow network as the product mix changes. Intuitively, a shop with movable machines and a virtual flow network is seen to have the highest freedom for reconfiguration, so that the total material handling distance could be the least.

However, the task is not as easy as it looks. One of the biggest issues is the order of machine assignment and network design. Suppose one decides to design the flow network first and then assign machines to slots. Without knowing the locations of machines, the process of designing a flow network is meaningless and invalid.

The alternative is to assign machines to slots first and then design the flow network. However, the entire process is still awkward. Consider a 12-machine shop in which the QAP technique is employed for machine assignment. Because an associated flow network has not been designed, the required distance information between slots in the QAP technique might be substituted by using the rectilinear measure.

As shown in Figure 65, the twelve machines are assigned to slots for some particular production session. The shortest distance from Machine 3 to Machine 9 is 20 distance units (5+10+5 = 20), represented by bold lines in Figure 65. Thereafter, suppose an associated flow network might be designed and obtained as shown in Figure 66. Note here, the shortest distance from Machine 3 to Machine 9 changes to 50 distance units (5+5+5+10+5+5+10+5=50), represented by bold lines in Figure 66.

Obviously, the shortest path to link Machine 3 to Machine 9 in machine assignment (Figure 65) is not the same as in network design (Figure 66). Because of the inconsistency, the entire design process is awkward. In practice, the problem of considering both movable machines and a virtual flow network in a shop is very complicated and is itself a research topic. Hence, in this study, the discussion is focused on the performance comparisons of a

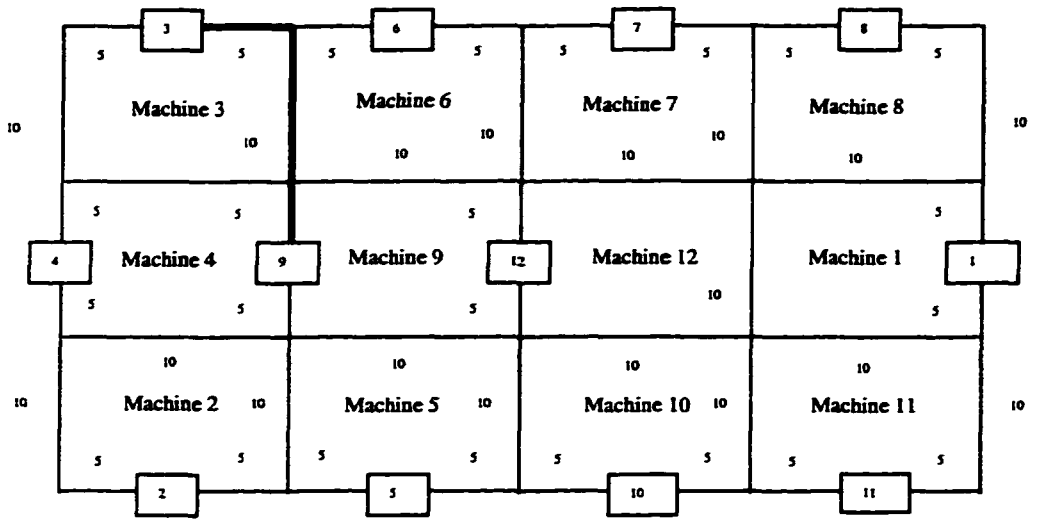


Figure 65. Assignment of machines to slots

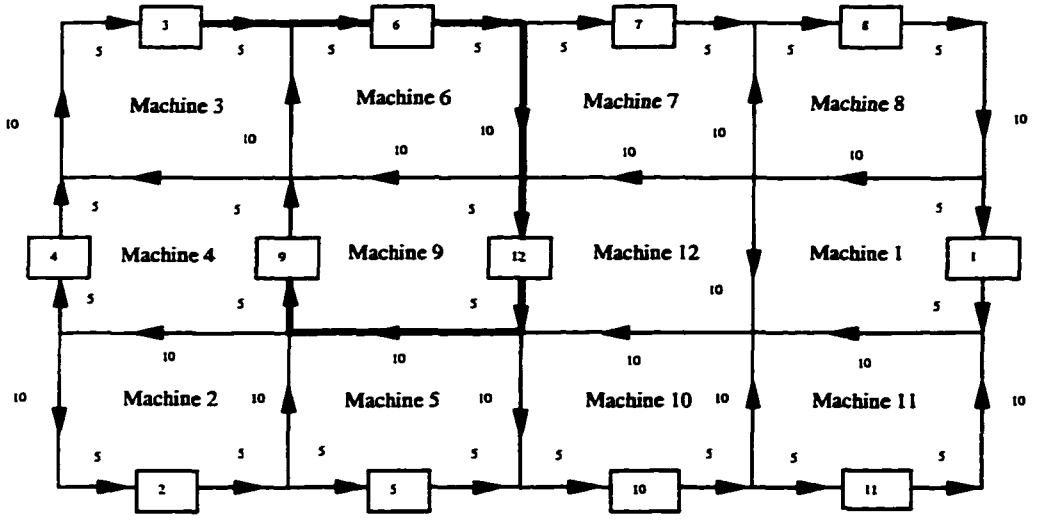


Figure 66. Design of an associated flow network

shop with either movable machines or a virtual flow network.

7.6 Results and Conclusions

The comparisons in this section fall into two categories. In the first category, a shop with movable machines is assumed and the performance of the shop under different operation modes is investigated. In the second category, the performance is compared for a variety of operation modes with different combinations of machines (fixed or movable) and flow networks (fixed or virtual).

In the first category, a shop is configured as a job shop type and then as a cellular type. For the cellular type organization, two cell formation methods are employed, as described in Section 7.3. The five examples in Chapter 6 are employed and the results obtained for the three systems (1 job shop type and 2 cellular types) are presented.

As shown in Table 81, the comparison is based on three measures, total setup time (Setup), total material handling distance (MH Dist.), and weighted performance value (Weighted). The formula in Section 6.5 for computing weighted performance value is applied here. A subtotal value for each example and a total value for the five examples are calculated for the three measures. Also shown at the bottom of Table 81 are the percentage data and the number of best solutions derived from each system. The percentage data are computed by using the results from the job shop as the base data and the corresponding measures for the two cellular configurations. For example, the 66.1% shown at the bottom of column 5 is obtained by dividing the Setup value in column 5 by the Setup value in column 2 (i.e., 8083.1/12235.0=0.661). Other percentage values are similarly computed.

The figure representing on the number of best solutions indicates the number of times the solutions obtained in one system outperform the solutions obtained in the other systems for the same measure of performance. If the obtained solutions of the three systems are tied for the same performance measure, the number of best solutions is increased by one for all three. For example, of the 50 production sessions, if based on the weighted performance value, 32 of the best solutions are obtained under the job shop type organization, while 10 and 8 are produced under the modified Boctor's model and Ko's method, respectively.

Table 81. The performance table of a shop with movable machines

		Job Shop Type			Modified Boctor's Model			Ko's Formation Approach		
		Setup	MH Dist.	Weighted	Setup	MH Dist.	Weighted	Setup	MH Dist.	Weighted
Example 1	Prod. Session 1	308.0	78150	8123.0	198.6	78150	8013.6	201.3	90250	9226.3
	Prod. Session 2	331.0	71650	7496.0	221.2	84200	8641.2	231.6	84200	8651.6
	Prod. Session 3	302.0	82400	8542.0	124.1	98200	9944.1	158.9	88300	8988.9
	Prod. Session 4	362.0	135700	13932.0	147.1	149650	15112.1	180.9	135700	13750.9
	Prod. Session 5	365.0	134450	13810.0	213.6	178500	18063.6	239.2	137450	13984.2
	Prod. Session 6	205.0	65900	6795.0	118.2	77500	7868.2	144.0	77500	7894.0
	Prod. Session 7	276.0	70000	7276.0	138.5	77900	7928.5	167.9	76800	7847.9
	Prod. Session 8	302.0	104200	10722.0	177.2	122600	12437.2	183.8	123400	12523.8
	Prod. Session 9	441.0	323550	32796.0	239.4	344550	34694.4	308.2	343350	34643.2
	Prod. Session 10	448.0	261900	26638.0	305.0	330250	33330.0	300.5	324050	32705.5
	SubTotal	3340.0	1327900	**136130.0**	1882.9	1541500	**156032.9**	2116.3	1481000	**150216.3**
Example 2	Prod. Session 1	206.0	344430	34649.0	155.6	369740	37129.6	150.1	369740	37124.1
	Prod. Session 2	241.0	346330	34874.0	179.2	379870	38166.2	251.3	419850	42236.3
	Prod. Session 3	162.0	177080	17870.0	151.2	200450	20196.2	184.3	215120	21696.3
	Prod. Session 4	286.0	724500	72736.0	187.2	864500	86637.2	289.0	864380	86727.0
	Prod. Session 5	179.0	172870	17466.0	116.0	172870	17403.0	146.8	189420	19088.8
	Prod. Session 6	124.0	74400	7564.0	126.7	74880	7614.7	160.1	82020	8362.1
	Prod. Session 7	357.0	495760	49933.0	233.0	610700	61303.0	312.9	616440	61956.9
	Prod. Session 8	179.0	234780	23657.0	118.5	234780	23596.5	95.1	234780	23573.1
	Prod. Session 9	215.0	408920	41107.0	137.0	480520	48189.0	174.4	508920	51066.4
	Prod. Session 10	328.0	832100	83538.0	201.7	991340	99335.7	298.9	978070	98105.9
	SubTotal	2277.0	3811170	**383394.0**	1606.1	4379650	**439571.1**	2062.9	4478740	**449936.9**
Example 3	Prod. Session 1	154.0	122324	12386.4	122.3	136426	13764.9	131.2	136426	13773.8
	Prod. Session 2	150.0	70374	7187.4	145.8	86059	8751.7	128.4	71699	7298.3
	Prod. Session 3	256.0	168867	17142.7	139.7	202398	20379.5	182.3	173368	17519.1
	Prod. Session 4	136.0	42058	4341.8	102.1	57629	5865.0	99.7	46005	4700.2
	Prod. Session 5	328.0	187066	19034.6	126.8	219984	22125.2	177.4	219984	22175.8
	Prod. Session 6	202.0	88589	9060.9	96.2	100989	10195.1	104.6	100989	10203.5
	Prod. Session 7	228.0	299577	30185.7	169.6	299577	30127.3	212.9	316149	31827.8
	Prod. Session 8	246.0	171089	17354.9	145.4	198622	20007.6	179.6	217956	21975.2
	Prod. Session 9	201.0	70326	7233.6	105.5	71635	7269.0	142.6	71635	7306.1
	Prod. Session 10	302.0	231933	23495.3	145.7	237350	23880.7	174.6	233548	23529.4
	SubTotal	2203.0	1452203	**147423.3**	1299.0	1610669	**162365.9**	1533.3	1587759	**160309.2**
Example 4	Prod. Session 1	207.0	94380	9645.0	166.7	116371	11803.8	210.3	108214	11031.7
	Prod. Session 2	221.0	72191	7440.1	187.6	86969	8884.5	230.1	87251	8955.2
	Prod. Session 3	228.0	78324	8060.4	181.9	96955	9877.4	221.9	87329	8954.8
	Prod. Session 4	258.0	137734	14031.4	205.2	143408	14546.0	246.5	183335	18580.0
	Prod. Session 5	199.0	101534	10352.4	127.3	118394	11966.7	138.7	113260	11464.7
	Prod. Session 6	258.0	100078	10265.8	195.0	116706	11865.6	279.3	142952	14574.5
	Prod. Session 7	200.0	43988	4598.8	135.2	44002	4535.4	155.2	46374	4792.6
	Prod. Session 8	414.0	153032	15717.2	257.8	160316	16289.4	370.6	181311	18501.7
	Prod. Session 9	223.0	65211	6744.1	165.0	100338	10198.8	200.5	75637	7764.2
	Prod. Session 10	196.0	40962	4292.2	122.8	42314	4354.2	142.2	48507	4992.9
	SubTotal	2404.0	887434	**91147.4**	1744.6	1025773	**104321.9**	2195.3	1074170	**109612.3**
Example 5	Prod. Session 1	179.0	529800	53159.0	159.1	529800	53139.1	142.6	609300	61072.6
	Prod. Session 2	250.0	1048650	105115.0	155.7	1144350	114590.7	209.1	1189300	119139.1
	Prod. Session 3	197.0	959900	96187.0	155.7	1218450	122000.7	178.9	1156300	115808.9
	Prod. Session 4	189.0	835250	83714.0	162.7	1033950	103557.7	162.0	1003550	100517.0
	Prod. Session 5	204.0	816300	81834.0	150.8	910550	91205.8	154.7	927450	92899.7
	Prod. Session 6	245.0	1308300	131075.0	186.9	1463350	146521.9	197.7	1423550	142552.7
	Prod. Session 7	206.0	1257950	126001.0	172.0	1424300	142602.0	186.8	1424300	142616.8
	Prod. Session 8	118.0	604250	60543.0	109.9	711300	71239.9	106.9	604250	60531.9
	Prod. Session 9	157.0	552950	55452.0	102.4	553050	55407.4	148.3	606150	60763.3
	Prod. Session 10	266.0	1303550	130621.0	195.3	1393100	139505.3	228.8	1390750	139303.8
	SubTotal	2011.0	9216900	**923701.0**	1550.5	10382200	**1039770.5**	1715.8	10334900	**1035205.8**
	Total	12235.0	16695607.0	1681795.7	8083.1	18939792.0	1902062.3	9623.6	18956569.0	1905280.5
	%	100.0%	100.0%	100.0%	66.1%	113.4%	113.1%	78.7%	113.5%	113.3%
	# of Best Solutions	1	50	32	42	5	10	7	3	8

In the measure of Setup, the percentages are 100%, 66.1%, and 78.7% for the job shop type, the modified Boctor's model, and Ko's method, respectively. As described in Section 6.3, because of machine grouping and part families, the two cellular type systems are superior to the job shop type system if Setup time is used as a measure. However, the two cellular type systems produced significantly different results in this measure. The reason for the difference in performance between the two cellular type systems is that machine sharing is not as significant in the modified Boctor's model as in Ko's method. Therefore, the Setup value for Ko's method is higher than for the modified Boctor's model.

In the measure of MH Dist., the percentage data are 100%, 113.4%, and 113.5% for the job shop type, the modified Boctor's model, and Ko's method, respectively, as shown in Table 81. Based on their organization types, the job shop type system has the highest freedom for machine assignment; however, machine assignment is restricted by considering cell configuration for the cellular type systems. Therefore, the job shop type system dominates the two cellular type systems in the material handling distance measure. Otherwise, the two cellular type systems perform very similarly with regard to the MH Dist.

Moreover, the values of Setup and MH Dist. are integrated in weighted performance value. As shown in Table 81, the percentage data of weighted performance value are 100%, 110.2%, and 111.2% for the job shop type, the modified Boctor's model, and Ko's method, respectively. Although the job shop type system is inferior to the two cellular type systems with regard to Setup, it outperforms its two counterparts with regard to MH Dist. Because the measure of MH Dist. governs the measure of Setup in weighted performance value (as mentioned in Section 6.5) the job shop type system produces the best performance in the measure of weighted performance value.

In the second category, the difference between a shop with and without movable machines is examined. The information of Tables 77 and 80 is summarized in Table 82, in which, nine different system configurations are presented. Each system is assigned a name based on its characteristics. For example, JS/VN represents a system that is organized as a job shop type and that has a virtual flow network. Note that, because the focus is on integrating both Setup and MH Dist., weighted performance value is the only measure presented in Table 82.

Table 82. The comparison among nine systems

	Fixed Machine			Fixed Machine			Movable Machine		
	Fixed Network			Virtual Network			Fixed Network		
	JS/FN	MC/FN	VC/FN	JS/VN	MC/VN	VC/VN	JS	Boctor's	Ko's
Example 1	172790.0	171387.4	171566.3	156200.0	154797.4	154976.3	136130.0	156032.9	150216.3
Example 2	647257.0	646621.9	647042.9	499793.0	499157.9	499578.9	383394.0	439571.1	449936.9
Example 3	179492.9	178491.6	178823.2	167322.5	166321.2	166652.8	147423.3	162365.9	160309.2
Example 4	194491.3	193865.0	194282.6	135670.4	135044.1	135461.7	91147.4	104321.9	109612.3
Example 5	1497036.0	1496550.2	1496740.8	910386.0	909900.2	910090.8	923701.0	1039770.5	1035205.8
Total	2691067.2	2686916.1	2688455.8	1869371.9	1865220.8	1866760.5	1681795.7	1902062.3	1905280.5
%	100.0%	99.8%	99.9%	69.5%	69.3%	69.4%	62.5%	70.7%	70.8%

Keys:
1. FN: fixed network
2. VN: virtual network
3. JS: job shop
4. MC: machine cell formed by using Boctor's model
5. VC: virtual cell formed by using Ko's method
6. Boctor's: machine cell formed by using the modified Boctor's model
7. Ko's: machine cell formed by using Ko's method

Also shown at the bottom of Table 82 are the percentage data, which are computed by using the results obtained with the JS/FN configuration as the base system. For instance, the value 69.3% under MC/VN is obtained by dividing the value in the MC/VN column by the value in the JS/FN column (1865220.8/2691067.2 = 0.693).

As shown in Table 82, the nine systems fall into three major categories: fixed machine and fixed network, fixed machine and virtual network, and movable machine and fixed network. The three systems in the first category (JS/FN, MC/FN, and VC/FN) are inferior to the systems in the other two categories. The systems in the last two categories try to reduce total material handling distance by considering either virtual flow network or movable machines, while the three systems in the first category retained their inflexibility by keeping both the machine location and network layout fixed. As shown in Table 82, the traditional inflexible systems have the worst performance of all scenarios tested.

The comparison between production systems with either a virtual flow network or movable machines is more interesting. As shown in Table 82, a job shop with movable

machines outperforms all the other systems, with a weighted measure of 62.5% of the JS/FN system. This is because it has the highest freedom to reorganize itself in response to changes in product mixes, compared with the other systems. On the other hand, the percentage performance level of a job shop with a virtual flow network, JS/VN, is about 69.5%. The results indicated that a job shop type system with movable machines and a fixed network is superior to a job shop type system with fixed machines and a virtual flow network. However, JS/VN performs far better than any system that involves fixed machines and a fixed network, as shown in Table 82.

For cellular type systems, the use of either a virtual flow network or movable machines seems to result in similar performance. As shown in Table 82, the percentage performance relative to the JS/FN system for MC/VN, VC/VN, Boctor's, and Ko's method are 69.3%, 69.4%, 70.7%, and 70.8%, respectively. Although the performance of the four cellular type systems are close to each other, the two cellular type systems with virtual flow networks (MC/VN and VC/VN) slightly outperformed the two cellular type systems with movable machines (Boctor's and Ko's). The reason might be that the cellular systems with a virtual flow network are granted more design freedom than the cellular systems with movable machines.

The impact of movable machines on a shop is significant, as shown in Table 82; a job shop type system with movable machines produces the best performance among all nine systems. The results show that, for cellular type systems, a policy that allows machines to be moved while retaining a fixed network competes favorably with the policy in shops with fixed machines and virtual flow network.

However, the adoption of movable machines implies physical rearrangement of a shop, and this might incur some additional cost. Besides, in the real world, few manufacturing shops are sufficiently flexible to permit frequent rearrangement of the machines. Although the use of movable machines has its advantages, it is not feasible for most manufacturers.

In contrast, the effect of a virtual flow network is similar to the effect of movable machines. The cost of redesigning a flow network is smaller than that of physically reorganizing a shop. When considering the fact that machines are unmovable in most

industries, a virtual flow network provides an alternative to improve the efficiency of production.

Based on the results reported in this chapter, several conclusions can be made:

(1) The impact of movable machines on a shop is significant. By physically rearranging a shop in response to product mix changes, efficiency could be improved.

(2) Systems with either a virtual flow network or movable machines perform better than traditional systems.

(3) A job shop type system with movable machines has the highest freedom to reorganize its machines and therefore produces the best performance among all systems investigated.

(4) A cellular type system with a virtual flow network performs slightly better than a cellular type system with movable machines. With cell configuration considered in machine assignment, a cellular system with movable machines has less design freedom to arrange its shop than a cellular system with a virtual flow network.

(5) A virtual flow network provides performance effects similar to the effects of movable machines. Considering the cost of machine reorganization and the fact that machines are usually locked in position, the virtual flow network provides an alternative to improve the efficiency of production.

CHAPTER 8. SUMMARY

This chapter summarizes the work done in this research. The chapter is organized into four topics: summary, contribution, future research, and conclusion.

8.1 Summary

The goal of the research was to develop a systematic procedure that allows an existing batch manufacturing shop to adapt its mode of operation in response to product mix changes. A production environment consisting of two modules, namely, a processing system configuration module and a networking module. The processing system configuration module considers three options (job shop configuration, traditional cellular configuration, and virtual cellular configuration), while the networking module considers two options (traditional flow network and virtual flow network). Combinations of the two modules produce six different operation modes. Of the six modes, four (JS/VN, MC/VN, VC/VN, and VC/FN) are considered virtual production systems. The other two modes (JS/FN and MC/FN) are considered traditional production systems. The six types of production system were observed in a dynamic changing product mix environment and examined by using three performance measures: total material handling distance, total setup time, and weighted performance value.

Material handling distance is one of the most important factors in a shop's performance. The shorter the total material handling distance, the more preferred a design. Because of the redesign of their flow networks in response to product mix changes, performance was superior for systems with virtual flow networks (JS/VN, MC/VN, and VC/VN) than for those with fixed flow networks (JS/FN, MC/FN, and VC/FN).

Setup time is another factor in the efficiency of a shop. The lower the total setup time, the greater the efficiency. Because machines and parts are grouped into machine cells and part families, the cellular type systems have an advantage in total setup time. According to the results, the cellular type systems (MC/FN, VC/FN, MC/VN, and VC/VN) required much less total setup time than the job shop type systems (JS/FN and JS/VN).

Weighted performance value, which integrates both total material handling distance and total setup time, is the other measure employed in evaluating the performance of a shop. The results showed that total material handling distance, rather than total setup time, dominates the measure of weighted performance value. Therefore, the system types that have lower total material handling distance perform better. Thus, those production systems that generate lower material handling time also produce lower weighted performance level values.

In general, virtual production systems can produce better performance levels than traditional production systems because they can respond to changing product mix. In addition, because physical reconfiguration is not required in a virtual production system, it is unnecessary to operate a shop on a fixed type of operation mode. Based on the design requirements of a virtual production system, a shop can always switch to one of the four virtual production modes and gain improved performance in its operation.

In another aspect of the research, the constraint of unmovable machines was latter relaxed by allowing physical reorganization of a shop to respond to changes in product mixes. The reconfigurable and movable machine systems were then analyzed under three production environments; one is a job shop and the other two are versions of cellular manufacturing systems. In each case, a flow network is assumed to exist and is fixed. The three operation modes with movable machines were compared with the previous six operation modes by using the same product mix data and performance measures.

The operation modes with movable machines were superior to traditional production systems and competitive to virtual production systems. Furthermore, a job shop type production system with movable machines outperformed the two modes of cellular manufacturing systems because it had the most freedom for machine rearrangement. However, if a shop is organized as a cellular type system and operated using a virtual network, its performance level is comparable to that of a job shop with movable machines. In practice, the cellular production systems with virtual flow networks (MC/VN and VC/VN) were slightly better than the cellular production systems with movable machines (Boctor's and Ko's).

In general, the use of both virtual flow networks and movable machines have similar effects on total material handling distance reduction. However, considering the cost of

physically reconfiguring a shop and the fact that movable machines are not practical for most companies, usage of virtual flow networks provide a feasible and reasonable means to improve a shop's performance.

8.2 Contributions

The key contributions of this research can be summarized as follows:

(1) The idea of a virtual production system is proposed. In a dynamic changing product mix environment, ability to adapt to changes to improve production efficiency is desirable. In this research, models are developed for designing virtual production systems that respond to changes in product mix.

(2) A virtual cell formation algorithm is proposed. Unlike traditional cell formation methods, the virtual cell formation approach allows for sharing of machines between cells. By taking the data of job routings and demands as inputs, the proposed algorithm generates virtual cells without specifying any artificial parameters.

(3) An AGV guidepath network design algorithm is proposed. The design of a flow network directly affects the total material handling distance incurred. Material handling distance impacts a shop's performance significantly. With a distance matrix file and a flow volume file as inputs, the proposed algorithm generates a near optimal flow network within a reasonable time.

(4) A machine cell location procedure is proposed. For cellular type systems, machines belonging to the same cell are considered for location assignment at the same time. Given a flow network, the proposed algorithm allocates machine cells to the shop floor so as to make it possible to reduce total material handling distance.

(5) The nine production system types (two traditional types, four virtual types, and three types with movable machines) were examined in terms of three performance measures: total setup time, total material handling distance, and weighted performance value. Using these performance measures, the relative effectiveness was compared for the nine production systems. The results provide valuable insights to system designers.

8.3 Future Research

The virtual production system is proposed as a solution for a shop whose layout cannot be physically reorganized in response to product mix changes to improve its performance/efficiency. Although the performance of a virtual production system has been demonstrated in the study, several extensions are possible for future research:

(1) One possible extension is to relax the constraint that each machine type has only one workstation in a shop. With the relaxation, more detailed observations are required and the design process of the virtual production system needs some modifications.

(2) A measure for evaluating the quality of design of virtual cells is required. Although several measures have been presented in the literature for use in evaluating traditional machine cells, these measures are not suitable for evaluating virtual cells because of the machine sharing concept. There is a need to develop such measures for virtual cells.

(3) The proposed AGV guidepath network design has been shown to produce good flow networks. However, the optimality of the created flow network cannot be guaranteed. So far, mathematical models available for dealing with flow network design work only for small problems (less than 10 machines), and the required computation time is high. A computationally effective method is required to evaluate the quality of a network design for systems with a large number of machines.

(4) Of all the systems evaluated, the most effective production systems were those that operate with either fixed machines with a virtual flow network, or movable machines with a fixed flow network. A production system that simultaneously allows for movable machines and a virtual flow network is desirable for investigation. In this study, such a production system was not evaluated because there is no algorithm currently available for its design. A design algorithm for shops with movable machines and a virtual flow network is needed.

(5) The structure of the proposed virtual production system could be extended by considering a variety of other features. A scheduling feature might need to schedule jobs on machines and/or cells, so as to minimize the incurred total flow time, tardiness cost, etc. In addition, consideration of jobs with alternative routing would be a good extension for future research.

(6) Visualization is another interesting aspect that can be incorporated into the research. It would be very beneficial if a virtual production system could be visualized through computer animation before being implemented in a shop, so that a manufacturer could visualize the system layout, machine cells, and the flow network.

8.4 Conclusion

To survive in today's customer-oriented market, a shop must be able to adapt itself so as to satisfy a variety of customer demands. For this purpose, virtual production system has been proposed to improve a shop's performance by allowing it to switch its operation type from one mode to another in response to changes in product mix. The performance of a virtual production system has been demonstrated through examples in this thesis. The major beauty of a virtual production system is that it provides a feasible and reasonable means of improving a shop's performance in a changing product mix environment without involving physical reconfiguration of the shop layout. It is believed that virtual production systems can benefit many industrial manufacturers.

APPENDIX A. AN EXAMPLE OF CREATING A CANDIDATE CELL

For the sake of illustration, the first candidate first machine (machine 1) and the first candidate last machine (machine 3) in Table 6 are used as an example to demonstrate the operation of the candidate cell creation procedure in Section 3.6. By having one candidate first machine and one candidate last machine, the operation starts from Step 4:

Step 4:

- 4.a. Store all machines in U.
  $U = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18\}$
- 4.b. Ignore all candidate first machines and candidate last machines, except machines 1 and 3. $U = \{1, 2, 3, 4, \underline{5}, 6, 7, 8, \underline{9}, \underline{10}, 11, 12, \underline{13}, 14, 15, 16, 17, 18\}$
- 4.c. Set $T = \{\}$
- 4.d. Set Set_flag = 0;
- 4.e. Store machine 1 in T. $T = \{1\}$
- 4.f. Deactivate machine 1 in U.
  $U = \{\underline{1}, 2, 3, 4, \underline{5}, 6, 7, 8, \underline{9}, \underline{10}, 11, 12, \underline{13}, 14, 15, 16, 17, 18\}$
- 4.g. Set $TL = \{0\}$
- 4.h. Set $C = \{\}$
- 4.i. Store machine 1 in C. $C = \{1\}$

Step 5:

- 5.a. machine 2 is identified
- 5.b. Store machine 1 in TL. $TL = \{0, 1\}$
- 5.c. Set $C = \{\}$
- 5.d. Store machine 2 in C. $C = \{2\}$
- 5.e. Store machine 2 in T. $T = \{1, 2\}$
- 5.f. Deactivate machine 2 in U.
  $U = \{\underline{1}, \underline{2}, 3, 4, \underline{5}, 6, 7, 8, \underline{9}, \underline{10}, 11, 12, \underline{13}, 14, 15, 16, 17, 18\}$

Step 6:

Because there is one machine in C and the machine is not the candidate last machine, machine 3, Step (6.c) is activated. Go to Step 5.

Step 5:

- 5.a. machine 3 is identified
- 5.b. Store machine 2 in TL. $TL = \{0, 1, 2\}$
- 5.c. Set $C = \{\}$
- 5.d. Store machine 3 in C. $C = \{3\}$
- 5.e. Store machine 3 in T. $T = \{1, 2, 3\}$
- 5.f. Deactivate machine 3 in U.
  $U = \{\underline{1}, \underline{2}, \underline{3}, 4, \underline{5}, 6, 7, 8, \underline{9}, \underline{10}, 11, 12, \underline{13}, 14, 15, 16, 17, 18\}$

Step 6:

Because there is one machine in C and the machine is the candidate last machine, machine 3, Step (6.a) is activated. Therefore, set Set_flag = 1 and go to Step 7.

Step 7:

- 7.a. Obtain the candidate cell with $T = \{1, 2, 3\}$ and $TL = \{0, 1, 2\}$:

7.a.1 Set CC = []
7.a.2 Cell_member = 0
7.a.3 Extract machine 3 from T. T = {1, 2}
7.a.4 Set Current_machine = 3
7.a.5 Store machine 3 in CC and increase Cell_member by one.
CC = [3] and Cell_member = 1.
7.a.6 Identify the machine that directly precedes the Current_machine, machine 3, in TL. TL = {0, 1, 2}
7.a.7 machine 2 is not machine 0 and go to Step (7.a.8).
7.a.8 Extract machine 2 from T. T = {1}
Set Current_machine = 2 and go to Step (7.a.5).
7.a.5 Store machine 2 in CC and increase Cell_member by one.
CC = [2, 3] and Cell_member = 2
7.a.6 Identify the machine that directly precedes the Current_machine, machine 2, in TL. TL = {0, 1, 2}
7.a.7 machine 1 is not machine 0 and go to Step (7.a.8).
7.a.8 Extract machine 1 from T. T = { }
Set Current_machine = 1 and go to Step (7.a.5).
7.a.5 Store machine 1 in CC and increase Cell_member by one.
CC = [1, 2, 3] and Cell_member = 3.
7.a.6 Identify the machine that directly precedes the Current_machine, machine 1, in TL. TL = {0, 1, 2}
7.a.7 machine 0, output CC and Cell_member
CC = [1, 2, 3] and Cell_member = 3.

7.b. Increase Number_cell by one.

Step 8: Continue...

With machine 1 as the first machine and machine 3 as the last machine, a candidate cell is generated and denoted as [m1, m2, m3]. The procedure will continue to use another candidate first machine and candidate last machine pair, and try to produce an associated candidate cell by using the pair. The procedure will terminate after all pairs have been evaluated.

APPENDIX B. THE OPERATION OF THE KO'S METHOD

The example in Figure 13 is used to demonstrate how to operate the Ko's virtual cell formation procedure in Section 3.6.

ITERATION 1

Step 1: Input data (in Figure 13)

Step 2: From-To Table

Parts	From	To	Volume
part 1	0	9	200
part 1	9	7	200
part 1	7	8	200
part 1	8	5	200
part 1	5	4	200
part 1	4	18	200
part 1	18	5	200
part 1	5	6	200
part 1	6	10	200
part 1	10	1	200
part 1	1	2	200
part 1	2	3	200
part 1	3	0	200
part 2	0	11	150
part 2	11	10	150
part 2	10	12	150
part 2	12	7	150
part 2	7	13	150
part 2	13	14	150
part 2	14	15	150
part 2	15	16	150
part 2	16	17	150
part 2	17	1	150
part 2	1	2	150
part 2	2	3	150
part 2	3	0	150
part 3	0	9	325
part 3	9	7	325
part 3	7	8	325
part 3	8	5	325
part 3	5	11	325
part 3	11	10	325
part 3	10	12	325
part 3	12	7	325
part 3	7	13	325
part 3	13	14	325
part 3	14	15	325
part 3	15	16	325
part 3	16	17	325
part 3	17	0	325
part 4	0	9	405
part 4	9	7	405
part 4	7	8	405
part 4	8	5	405
part 4	5	4	405
part 4	4	18	405
part 4	18	5	405
part 4	5	6	405
part 4	6	10	405
part 4	10	13	405
part 4	13	14	405
part 4	14	15	405
part 4	15	16	405
part 4	16	17	405
part 4	17	1	405
part 4	1	2	405
part 4	2	3	405
part 4	3	0	405

Nondecreasing-To Table

Parts	From	To	Total
part 1	10	1	200
part 2	17	1	555
part 1	1	2	755
part 1	2	3	755
part 1	5	4	605
part 1	8	5	930
part 1	18	5	605
part 1	5	6	605
part 1	9	7	930
part 2	12	7	475
part 1	7	8	930
part 1	6	10	605
part 2	11	10	475
part 3	5	11	325
part 2	10	12	475
part 2	7	13	475
part 4	10	13	405
part 2	13	14	880
part 2	14	15	880
part 2	15	16	880
part 2	16	17	880
part 1	4	18	605

Nondecreasing-From Table

Parts	From	To	Total
part 1	1	2	755
part 1	2	3	755
part 1	4	18	605
part 1	5	4	605
part 1	5	6	605
part 3	5	11	325
part 1	6	10	605
part 1	7	8	930
part 2	7	13	475
part 1	8	5	930
part 1	9	7	930
part 1	10	1	200
part 2	10	12	475
part 4	10	13	405
part 2	11	10	475
part 2	12	7	475
part 2	13	14	880
part 2	14	15	880
part 2	15	16	880
part 2	16	17	880
part 2	17	1	555
part 1	18	5	605

Step 3: In-degree, Out-degree, and Difference

machine	In-degree	Out-degree	Difference
1	2	1	1
2	1	1	0
3	1	0	1
4	1	1	0
5	2	3	-1
6	1	1	0
7	2	2	0
8	1	1	0
9	0	1	-1
10	2	3	-1
11	1	1	0
12	1	1	0
13	2	1	1
14	1	1	0
15	1	1	0
16	1	1	0
17	1	1	0
18	1	1	0

Step 4: Generate **candidate cells**

Candidate first machine	Candidate last machine
[1	]
[	3]
[	5]
[9	]
[	10]
[13	]

Step 4: Generate Candidate cells

cell 1: [1 2 3]
cell 2: [9 7 8 5]

Step 5: Evaluate Candidate cells

cell 1: [1 2 3]
cell 2: [9 7 8 5]

Step 6: Update the current job routings

part 1: 0-**cell 2**-4-18-5-6-10-**cell 1**-0
part 2: 0-11-10-12-7-13-14-15-16-17-**cell 1**-0
part 3: 0-**cell 2**-11-10-12-7-13-14-15-16-17-0
part 4: 0-**cell 2**-4-18-5-6-10-13-14-15-16-17-**cell 1**-0

part 1: 0-4-18-5-6-10-0
part 2: 0-11-10-12-7-13-14-15-16-17-0
part 3: 0-11-10-12-7-13-14-15-16-17
part 4: 0-4-18-5-6-10-13-14-15-16-17-0

Step 7: Go to Step 8.

Step 8: Not all exceptional machines; go to Step 2 with the updated job routings.

ITERATION 2

Step 2: From-To Table

Parts	From	To	Volume
part 1	0	4	200
part 1	4	18	200
part 1	18	5	200
part 1	5	6	200
part 1	6	10	200
part 1	10	0	200
part 2	0	11	150
part 2	11	10	150
part 2	10	12	150
part 2	12	7	150
part 2	7	13	150
part 2	13	14	150
part 2	14	15	150
part 2	15	16	150
part 2	16	17	150
part 2	17	0	150
part 3	0	11	325
part 3	11	10	325

Parts	From	To	Volume
part 3	10	12	325
part 3	12	7	325
part 3	7	13	325
part 3	13	14	325
part 3	14	15	325
part 3	15	16	325
part 3	16	17	325
part 3	17	0	325
part 4	0	4	405
part 4	4	18	405
part 4	18	5	405
part 4	5	6	405
part 4	6	10	405
part 4	10	13	405
part 4	13	14	405
part 4	14	15	405
part 4	15	16	405
part 4	16	17	405
part 4	17	0	405

Nondecreasing-To Table

Parts	From	To	Total
part 1	18	5	605
part 1	5	6	605
part 2	12	7	475
part 1	6	10	605
part 2	11	10	475
part 2	10	12	475
part 2	7	13	475
part 4	10	13	405
part 2	13	14	880
part 2	14	15	880
part 2	15	16	880
part 2	16	17	880
part 1	4	18	605

Nondecreasing-From Table

Parts	From	To	Total
part 1	4	18	605
part 1	5	6	605
part 1	6	10	605
part 2	7	13	475
part 2	10	12	475
part 4	10	13	405
part 2	11	10	475
part 2	12	7	475
part 2	13	14	880
part 2	14	15	880
part 2	15	16	880
part 2	16	17	880
part 1	18	5	605

Step 3: In-degree, Out-degree, and Difference

machine	In-degree	Out-degree	Difference
4	0	1	-1
5	1	1	0
6	1	1	0
7	1	1	0
10	2	2	0
11	0	1	-1
12	1	1	0
13	2	1	1
14	1	1	0
15	1	1	0
16	1	1	0
17	1	0	1
18	1	1	0

Step 4: Generate **candidate cells**

Candidate first machines	Candidate last machines
[4	]
[	11]
[13	]
[	17]

Step 4: Generate Candidate cells

cell 3: [13 14 15 16 17]

Step 5: Evaluate Candidate cells

cell 1: [1 2 3]
cell 2: [9 7 8 5]
cell 3: [13 14 15 16 17]

Step 6: Update the current job routings

part 1: 0-4-18-5-6-10-0
part 2: 0-11-10-12-7-**cell 3**-0
part 3: 0-11-10-12-7-**cell 3**-0
part 4: 0-4-18-5-6-10-**cell 3**-0

part 1: 0-4-18-5-6-10-0
part 2: 0-11-10-12-7-0
part 3: 0-11-10-12-7-0
part 4: 0-4-18-5-6-10-0

Step 7: Go to Step 8.

Step 8: Not all exceptional machines; go to Step 2 with the updated job routings.

ITERATION 3

Step 2: From-To Table

Parts	From	To	Volume
part 1	0	4	200
part 1	4	18	200
part 1	18	5	200
part 1	5	6	200
part 1	6	10	200
part 1	10	0	200
part 2	0	11	150
part 2	11	10	150
part 2	10	12	150
part 2	12	7	150
part 2	7	0	150

Parts	From	To	Volume
part 3	0	11	325
part 3	11	10	325
part 3	10	12	325
part 3	12	7	325
part 3	7	0	325
part 4	0	4	405
part 4	4	18	405
part 4	18	5	405
part 4	5	6	405
part 4	6	10	405
part 4	10	0	405

Nondecreasing-To Table

Parts	From	To	Total
part 1	18	5	605
part 1	5	6	605
part 2	12	7	475
part 1	6	10	605
part 2	11	10	475
part 2	10	12	475
part 1	4	18	605

Nondecreasing-From Table

Parts	From	To	Total
part 1	4	18	605
part 1	5	6	605
part 1	6	10	605
part 2	10	12	475
part 2	11	10	475
part 2	12	7	475
part 1	18	5	605

Step 3: In-degree, Out-degree, and Difference

Machine	**In**	**Out**	**Difference**
4	0	1	-1
5	1	1	0
6	1	1	0
7	1	0	1
10	2	1	1
11	0	1	-1
12	1	1	0
18	1	1	0

Step 4: Generate **candidate cells**

Candidate first machines	Candidate last machines
[4	]
[	7]
[	10]
[11	]

Step 4: Generate Candidate cells

cell 4: [4 18 5 6 10]
cell 5: [11 10]

Step 5: Evaluate Candidate cells

cell 1: [1 2 3]
cell 2: [9 7 8 5]
cell 3: [13 14 15 16 17]
cell 4: [4 18 5 6 10]
cell 5: [11 10]

Step 6: Update the current job routings

part 1: 0-**cell 4**-0
part 2: 0-**cell 5**-12-7-0
part 3: 0-**cell 5**-12-7-0
part 4: 0-**cell 4**-0

part 1: 0
part 2: 0-12-7-0
part 3: 0-12-7-0
part 4: 0

Step 7: Go to Step 8.
Step 8: Not all exceptional machines; go to Step 2.

ITERATION 4

Step 2: From-To Table

Parts	From	To	volume
part 2	0	12	150
part 2	12	7	150
part 2	7	0	150
part 3	0	12	325
part 3	7	0	325
part 3	12	7	325

Nondecreasing-To Table

parts	from	to	total
part 2	12	7	150

Nondecreasing-From Table

parts	from	to	total
part 2	12	7	150

Step 3: Calculate the Difference

machine	**In**	**Out**	**Difference**
7	1	0	1
12	0	1	-1

Step 4: Generate **candidate cells**

Candidate first machines	Candidate last machines
[	7]
[12	]

Step 4: Generate Candidate cells

cell 6: [12 7]

Step 5: Integrate Candidate cells

cell 1: [1 2 3]
cell 2: [9 7 8 5]
cell 3: [13 14 15 16 17]
cell 4: [4 18 5 6 10]
cell 5: [11 10]
cell 6: [12 7]

Step 6: Update the current job routings

part 1: **cell 2-cell 4-cell 1**
part 2: **cell 5-cell 6-cell 3-cell 1**
part 3: **cell 2-cell 5-cell 6-cell 3**
part 4: **cell 2-cell 4-cell 3-cell 1**

part 1: 0
part 2: 0
part 3: 0
part 4: 0

Step 7: Go to Step 8.

Step 8: All operational sequences of products have been replaced by candidate cells;, go to Step 9.

Step 9: Resolve the exceptional machines;, go to Step 10.

Step 10: Update all candidate cells and the job routings again

part 1: cell 2-cell 4-cell 1
part 2: cell 5-cell 3-cell 1
part 3: cell 2-cell 5-cell 3
part 4: cell 2-cell 4-cell 3-cell 1

Step 11: Terminate; the candidate cells on hand are the cells themselves, shown as Figure 7.

After being processed by the Ko's virtual cell formation procedure, the example in Figure 13 could be represented by using virtual cells as:

part 1: cell 2-cell 4-cell 1
part 2: cell 5-cell 3-cell 1
part 3: cell 2-cell 5-cell 3
part 4: cell 2-cell 4-cell 3-cell 1

APPENDIX C. AN EXAMPLE OF DIJKSTRA'S ALGORITHM

The procedure for using Dijkstra's Algorithm for finding the shortest path based on the example problem in Section 4.3 is presented:

Step 1: $T = \{v_1\}$

$W = \{v_0, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}, v_{11}, v_{12}, v_{13}\}$

$P = \{0\}$

$PL = \{0\}$

Step 2: Adjacent vertices: v_0

Temporary labels for $v_0 = 0 + 1 = 1$.

Step 3: Permanent label $v_0 = 1$.

$T = \{v_1, v_0\}$

$P = \{0, v_1\}$

$PL = \{0, 1\}$

$W = \{v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}, v_{11}, v_{12}, v_{13}\}$

Step 2: Adjacent vertices: v_2

Temporary labels for $v_2 = 1 + 2 = 3$

Step 3: Permanent label for $v_2 = 3$

$T = \{v_1, v_0, v_2\}$

$P = \{0, v_1, v_0\}$

$PL = \{0, 1, 3\}$

$W = \{v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}, v_{11}, v_{12}, v_{13}\}$

Step 2: Adjacent vertices: v_{10}

Temporary labels for $v_{10} = 3 + 1 = 4$

Step 3: Permanent label for $v_{10} = 4$

$T = \{v_1, v_0, v_2, v_{10}\}$

$P = \{0, v_1, v_0, v_2\}$

$PL = \{0, 1, 3, 4\}$

$W = \{v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{11}, v_{12}, v_{13}\}$

Step 2: Adjacent vertices: v_3, v_{11}

Temporary labels for $v_3 = 4$ (from v_{10}) + 2 = 6

Temporary labels for $v_{11} = 4$ (from v_{10}) + 2 = 6

Step 3: Permanent label for $v_3 = 6$

$T = \{v_1, v_0, v_2, v_{10}, v_3\}$

$P = \{0, v_1, v_0, v_2, v_{10}\}$

PL= {0, 1, 3, 4, 6}

$W = \{v_4, v_5, v_6, v_7, v_8, v_9, v_{11}, v_{12}, v_{13}\}$

Step 2: Adjacent vertices: v_{11}

Temporary labels for $v_{11} = 6$

Step 3: Permanent label for $v_{11} = 6$

$T = \{v_1, v_0, v_2, v_{10}, v_3, v_{11}\}$

$P = \{0, v_1, v_0, v_2, v_{10}, v_{10}\}$

PL= {0, 1, 3, 4, 6, 6}

$W = \{v_4, v_5, v_6, v_7, v_8, v_9, v_{12}, v_{13}\}$

Step 2: Adjacent vertices: v_5, v_{12}

Temporary labels for $v_5 = 6$ (from v_{11})+ 1 = 7

Temporary labels for $v_{12} = 6$ (from v_3) + 4 = 10

Step 3: Permanent label for $v_5 = 7$

$T = \{v_1, v_0, v_2, v_{10}, v_3, v_{11}, v_5\}$

$P = \{0, v_1, v_0, v_2, v_{10}, v_{10}, v_{11}\}$

PL= {0, 1, 3, 4, 6, 6, 7}

$W = \{v_4, v_6, v_7, v_8, v_9, v_{12}, v_{13}\}$

Since node 5 is included in *T*, the algorithm terminates. The shortest path is from v_1, v_0, v_2, v_{10}, v_{11}, to v_5, and the shortest distance is 7 units.

APPENDIX D. THE PRODUCT MIX DATA

The experiment has five examples, each with ten production sessions. The product mix in each session is as follows:

EXAMPLE 1

Production Sessions

	1		2		3		4		5	
Groups	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands
1	1 4 7 4 8 9 1 2 4 8 4 7 9 4 7 4 8	100 120 200 50	7 8 7 9 1 2 4 7 8 9 4 8 9 4 7 8 9 7 8	100 120 20 50 60	4 8 9 4 8 7 9 4 7 8 7 9 4 7 8 9	90 100 120 200				
2	3 5 2 6 4 8 9 3 5 7 8 2 6 3 5 3 5 4 7 8	90 80 70 75	3 5 4 8 9 2 3 4 5 8 9 3 5 6 4 8 9	90 80 75			6 7 8 6 8 7 6 8 7 8 7 9 6 7 9 3 5 7 9 8 9 3 5 4 7 8 3 4 3 5 3 4 7 8	100 120 150 80 75 150 80 190	3 5 4 9 3 5 7 8 3 5 4 7 8 3 5 4 8 9 4 7 8 7 9 4 7 9 3 4 8 9 3 4 3 5 8 9	90 80 190 110 75 100 110 170
3	1 1 7 1 2 1 1 1 2 1 1 7 1 0 1 1 1 2	70 50 70	6 8 1 0 1 2 3 6 8 1 0 1 2 3 8 1 0 1 2	70 50 70	1 1 1 2 8 9 7 1 7 8 9 1 7 9 1 1 1 7 1 1 1 2 1 0 1 1 1 2 1 1 7 1 2 1 7 1 7 1 1 1 2 1 1	75 90 70 50 70 90 100	1 7 1 1 1 2 1 1 7 1 2 1 0 1 1 1 2 1 2 1 1 7 1 1	110 60 80 90	1 6 7 1 1 1 1 2 1 6 7 9 7 1 1 1 2	80 75 150
4									1 6 4 7 1 6 1 1 1 2 1 7 9	100 150 120

Production Sessions

	6		7		8		9		10	
Groups	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands
1			1 4 8 9 7 8 7 9 4 7 4 8 4 7 9	120 110 105 90			1 4 8 9 1 4 7 4 7 9 1 2 4 7 8 9 4 9 4 7 8 4 8 7 9 4 7 8 7 9	200 300 100 300 200 210	4 7 8 9 4 7 4 8 7	200 200
2	3 5 7 8 3 5 7 9 8 9	90 80			5 3 4 8 9 3 4 5 4 8 7 3 5 3 5 4 3 4 8 9 7 8	80 90 40 200			3 5 4 7 4 8 9 3 5 4 8 9	100 100
3	1 6 1 0 7 9 6 1 0 7 8 9 6 7 1 0 1 6 7 1 0 1 0 1 1 1 2 7 1 1 1 2	100 130 110 90 190 200	1 2 1 6 1 0 1 6 7 1 0 1 2 6 7 1 0 7 1 2 1 1 1 2 1 2 1 1 7 1 1 1 1 7 1 0 1 1 1 2 1 7 1 1 1 2	70 80 60 50 45 50 60 70	1 6 7 1 6 7 9 1 6 7 1 0 1 6 9 7 8	70 80 100 200	1 6 1 0 7 9 6 1 0 7 8 9 1 1 7 1 2 6 7 1 0 1 5 7 8 1 7 8 9 1 6 1 0 1 2 6 7 8 1 1 7 1 2 1 7	200 100 100 300 75 50 80 300 140	1 1 7 1 2 1 1 1 2 1 1 7 1 0 1 7 1 1 0 1 1 1 2 3 5 1 6 1 0 8 9 1 6 1 0 1 2 6 1 6 1 6 7 1 0 1 2 6 1 1 7 1 2 1 7	100 100 300 300 45 80 300 225 200 140
4					4 7 1 1 4 7 9 1 2 1 1	100 70	4 8 9 1 5	250	1735 3 5 6 1 0	110 150

EXAMPLE 2

Production Sessions

	1		2		3		4		5	
Groups	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands
1	4 6 5 8 1 5 8 3 8 4 8 6 1 1 6 4 6	472 32 180 81 310 459			1 6 1 3 3 1 3	71 227 355	4 5 6 3 4 1 5 5 4 3 5 4 1 6 8 4 1 4 5 4 3	135 290 335 287 182		
2	2 3 7 6 4 2 4 2 7	182 71 72	4 2 6 6 7 2 7 6	276 32 111	3 7 4 6 2 4	97 36	4 2 6 6 4 6 4 3	304 276 225	2 7 7 3 7 6	336 172 410
3	5 2 4 4 1 4	272 142	8 7 6 7 5 2 4 7 1 1 5 2 4 4 6 7 8 2 5 1 7 1 6 7 5 7	14 187 39 11 305 55 404	2 1 8 2 8 6 7 5 8 4	330 205 341	2 6 4 1 4 1 6 2 7 1	395 462 334	8 2 4 6 7 1 5 2 6 2 4 5 6 5 8 6 4 1 5	304 11 300 185 260 108 75
4										
5			5 2 6 8 3 6 8 2	337 180			3 5 2 1 2 6 1 3	478 29		

	Production Sessions									
	6		7		8		9		10	
Groups	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands
1	8148 85 38 356	84 305 118 59	386 84863 563 8143 84 84361 351	444 101 302 127 116 21 124					53134 1818 35 51 361 86148	158 197 443 215 318 436
2										
3	162	48	17 142 41 61284	343 388 254 162			48 682 575 576 5836 56 5786	134 254 167 279 97 1 401		
4	8578 3245	85 72	278 283 58274 1273	74 72 380 166	7815 753 47 3747 341 138 71	303 5 3 456 249 354 32	8352 82 532 347 575	262 374 376 456 167	2427 5478 134 3471	191 108 334 277
5					852 24	289 4			562 53261 72631 512 468	371 186 199 459 373

EXAMPLE 3

	Production Sessions									
	1		2		3		4		5	
Groups	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands
1	232 1234	378 417	32 4312	442 206			32 43 24 232	251 439 268 54		
2	5154 54	432 471	241 4125 14 24	138 287 82 197	41 1425 1245 2524 1421	235 433 124 213 102			54 2142 152 452 15425 125	188 186 227 258 394 357
3					235 543 23514 4342 31 453 3134	130 308 395 487 78 224 179	43 53 1545 13 23	439 223 213 336 134	531 1432 5153 35 3435 23 52132 3254 23 45232	151 218 47 401 277 220 257 10 211 255
4	532 35 25231	266 481 371	52 31 32 525	188 57 442 349						

	Production Sessions									
	6		7		8		9		10	
Groups	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands
1	2143 24 14 3123 312 431	488 402 89 274 276 173			3432 21 1423 2434	324 476 437 87	231 314 1232 3124	467 29 128 165		
2			251 124	210 334	245 54	212 493			5241 51245 51 5125 15121	200 130 419 479 345
3	534 4515 515 34 53	103 73 163 230 60	14543 514 4352 52423 342343 3253	356 438 327 324 188 361	45353 5435 52121 12 315 13452	143 331 323 431 202 57	315 4235 35 15 52 1435	338 313 327 136 130 58	4153 34514 34 3514 43 14543	347 191 270 256 164 299
4			53 135 353 523 343 3215 315123	161 181 130 375 325 475 300					23515 31 2132	301 383 121

EXAMPLE 4

Production Sessions

	1		2		3		4		5	
Groups	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands
1	11 3 8 9 3 8 9	340 392 399	8 3 11 3	198 66	6 7 3 10	182	7 11 3 7	262		
2	1 4 9 11 10 5 10 4 10 6 11 7 10 7 9 10	120 102 437 204	10 7 4 8 7 10 8 9 8 10 7 1 7 10 6 8 6 8 1 11	240 295 109 172 113	7 5 10 10 7 6 7	180 496	1 4 1 10 7 5 8 6 10 6 10 6 1 10 10 9 9 8 10 4	43 243 303 397 460 349 131	10 4 7 8 7 1 1 10 9 10 7 8 4 7 4 1 6 10 8 10 5 5 8 11 4 8 6	213 288 385 267 385 78 360
3	9 4 1 3 2 4 2 4	254 370					3 2 1 2 2 10 3 1 3 11	456 339 310 109	11 2 4	114
4			7 5 3 6 8 1 8 5 1 7	144 400	3 5 1 6 3 2 7 8 5 4 3 1 6 4 8 5 7 2 3 9 5 3	328 404 195 386 77 295 426 71 64	3 5 9 6 4 5 4 5	362 455		

Production Sessions

	6		7		8		9		10	
Groups	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands
1	8 3 11 6	72			11 7 11 8 7 3 10 9 9 7 10 3 9 8 3 7	50 150 36 179	11 8	318		
2	2 1 4 7 11 5 6 5 1 11	286 189	5 13 8 5 4 7 8 6 8 2 9 11 2 6 9 4 9 6 2 8 5 1 5	203 267 53 406 187 27	5 11 5 10 6 2 6 8 9 5 7 5 10 11 9 10 2 7 5	152 177 264 45 242 232	11 1 7 8 7 5 9 7 12 10 6 9 8 5 7 5 2 7 4 2 4 2 8 7 2 8 1	105 193 317 164 177 176 284 24	4 10 7 9 5 1 5 4 8 4 1 10 5 7 2 5 5 2 5 2 8 7 2 9 6 7 2	324 45 72 205 253 126 185 388
3							3 11 5 2	414	3 11	324
4	2 1 4 8 2 7 9 6 7 4 8 6 3 4 5 3 4 8 7 9 8 3 1 8	311 200 372 456 193 438 413	3 1 8 2 8	118 76	8 4 9 2 6 8 7 8 6 3 4 9 4 6 4 8 2 9 8 9 5 8 5 2 7 7 1	82 313 341 399 54 287 156 131				

EXAMPLE 5

Production Sessions

	1		2		3		4		5	
Groups	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands
1	2 7 5 2 7 3 5 4 5	468 462 384	3 2 1 5 3 2 3 2 4 7 4 2 1 7 1	337 336 306 484 151	6 2 5 3 4 6 1 3 6 4 3 1 8	418 37 279	2 7 4 4 6 7 1 4	49 191 283	3 4 2 6 2 3 1 7 3 6 2 4 6	335 89 391 400
2	8 3 7 6 6 2 6 7 8	95 297 130 30	3 6 6 8	74 36			6 8 7 3 6 8 7 6 6 3	361 215 314 482	2 8 2 8 2 1 8 2 1 1 7 8 2	196 454 64 85
3	4 1 7 1 1 5	468 359 146	8 1 8 4 4 1 5 8 5 1 4 3 1 3 5 4 6	225 324 247 30 48	1 8 1 5 8 4 6 8 1 1 7 1 6	179 102 54 279	8 5 6 6 1 5 8 7 5	481 331 335	7 5 8 5 4	200 123
4					8 6 8 7 4 7	206				

Production Sessions

	6		7		8		9		10	
Groups	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands	Job Routings	Demands
1	4 6 1 3 2 3 1 7 6 5 1 2 6 4 6	457 232 338 99 376	4 1 4 6 5 7 5 3 5 6 5 1 6 5 1 6 2	208 277 425 406 463	6 1	10	6 5 2 4 6 3 6 4 6 3 3 4 6 2	307 483 457 76 300 137	4 1 5 1 2 4 7 4 2 7 1 4	76 93 136
2			8 2 1 8	87					7 6 8 3 8 3	87 79
3	8 1 5 3 3 5 7 8 5	375 473	4 7 8 7 8 4 7 3 4 3 8	426 453 331 475	3 8 5 8 4 8 5 8	247 361	8 1 8 3 4 6 5 8	204 304 142	8 4 6 8 6 5 5 4 3 1 8 1 4 1 8 1 4 3 5 8	244 395 409 249 69 237 468
4	8 2 5 2 5 8 2 5 8 2 8 7 5	166 283 376 230 248	8 4	263	2 1 4 6 8 1 8 2 1 8 1	240 260			8 4 8 2 8 2	473 479

APPENDIX E. AN EXAMPLE OF LINGO PROGRAM

The traditional cell formation is completed by LINGO. For instance, the LINGO program for Example 1, in which the number of machine cells and the largest number of machines in a cell are specified as 3 and 5, respectively, is presented as follows:

```
MIN= 100*(@ABS(X11-Y11)+@ABS(X12-Y12)+@ABS(X13-Y13))
     +120*(@ABS(X11-Y21)+@ABS(X12-Y22)+@ABS(X13-Y23))

     +120*(@ABS(X21-Y21)+@ABS(X22-Y22)+@ABS(X23-Y23))
     + 90*(@ABS(X21-Y51)+@ABS(X22-Y52)+@ABS(X23-Y53))
     + 70*(@ABS(X21-Y71)+@ABS(X22-Y72)+@ABS(X23-Y73))

     + 90*(@ABS(X31-Y51)+@ABS(X32-Y52)+@ABS(X33-Y53))
     + 80*(@ABS(X31-Y61)+@ABS(X32-Y62)+@ABS(X33-Y63))
     + 70*(@ABS(X31-Y71)+@ABS(X32-Y72)+@ABS(X33-Y73))
     + 75*(@ABS(X31-Y81)+@ABS(X32-Y82)+@ABS(X33-Y83))

     +200*(@ABS(X41-Y11)+@ABS(X42-Y12)+@ABS(X43-Y13))
     +120*(@ABS(X41-Y21)+@ABS(X42-Y22)+@ABS(X43-Y23))
     +200*(@ABS(X41-Y31)+@ABS(X42-Y32)+@ABS(X43-Y33))
     +100*(@ABS(X41-Y41)+@ABS(X42-Y42)+@ABS(X43-Y43))
     + 90*(@ABS(X41-Y51)+@ABS(X42-Y52)+@ABS(X43-Y53))
     + 75*(@ABS(X41-Y81)+@ABS(X42-Y82)+@ABS(X43-Y83))

     + 90*(@ABS(X51-Y51)+@ABS(X52-Y52)+@ABS(X53-Y53))
     + 80*(@ABS(X51-Y61)+@ABS(X52-Y62)+@ABS(X53-Y63))
     + 70*(@ABS(X51-Y71)+@ABS(X52-Y72)+@ABS(X53-Y73))
     + 75*(@ABS(X51-Y81)+@ABS(X52-Y82)+@ABS(X53-Y83))

     + 90*(@ABS(X61-Y51)+@ABS(X62-Y52)+@ABS(X63-Y53))
     + 70*(@ABS(X61-Y71)+@ABS(X62-Y72)+@ABS(X63-Y73))

     +100*(@ABS(X71-Y11)+@ABS(X72-Y12)+@ABS(X73-Y13))
     +200*(@ABS(X71-Y31)+@ABS(X72-Y32)+@ABS(X73-Y33))
     + 50*(@ABS(X71-Y41)+@ABS(X72-Y42)+@ABS(X73-Y43))
     + 80*(@ABS(X71-Y61)+@ABS(X72-Y62)+@ABS(X73-Y63))
     + 75*(@ABS(X71-Y81)+@ABS(X72-Y82)+@ABS(X73-Y83))
     + 70*(@ABS(X71-Y91)+@ABS(X72-Y92)+@ABS(X73-Y93))
     + 70*(@ABS(X71-Y111)+@ABS(X72-Y112)+@ABS(X73-Y113))

      +100*(@ABS(X81-Y11)+@ABS(X82-Y12)+@ABS(X83-Y13))
     +120*(@ABS(X81-Y21)+@ABS(X82-Y22)+@ABS(X83-Y23))
     + 50*(@ABS(X81-Y41)+@ABS(X82-Y42)+@ABS(X83-Y43))
     + 90*(@ABS(X81-Y51)+@ABS(X82-Y52)+@ABS(X83-Y53))
     + 80*(@ABS(X81-Y61)+@ABS(X82-Y62)+@ABS(X83-Y63))
     + 75*(@ABS(X81-Y81)+@ABS(X82-Y82)+@ABS(X83-Y83))

     +100*(@ABS(X91-Y11)+@ABS(X92-Y12)+@ABS(X93-Y13))
     +200*(@ABS(X91-Y31)+@ABS(X92-Y32)+@ABS(X93-Y33))
```

```
     + 90*(@ABS(X91-Y51)+@ABS(X92-Y52)+@ABS(X93-Y53))

     + 70*(@ABS(X101-Y111)+@ABS(X102-Y112)+@ABS(X103-Y113))

     + 70*(@ABS(X111-Y91)+@ABS(X112-Y92)+@ABS(X113-Y93))
     + 50*(@ABS(X111-Y101)+@ABS(X112-Y102)+@ABS(X113-Y103))
     +140*(@ABS(X111-Y111)+@ABS(X112-Y112)+@ABS(X113-Y113))

     + 70*(@ABS(X121-Y91)+@ABS(X122-Y92)+@ABS(X123-Y93))
     + 50*(@ABS(X121-Y101)+@ABS(X122-Y102)+@ABS(X123-Y103))
     + 70*(@ABS(X121-Y111)+@ABS(X122-Y112)+@ABS(X123-Y113));

X11+X21+X31+X41+X51+X61+X71+X81+X91+X101+X111+X121<=5;
X12+X22+X32+X42+X52+X62+X72+X82+X92+X102+X112+X122<=5;
X13+X23+X33+X43+X53+X63+X73+X83+X93+X103+X113+X123<=5;


X11+X12+X13=1;
X21+X22+X23=1;
X31+X32+X33=1;
X41+X42+X43=1;
X51+X52+X53=1;
X61+X62+X63=1;
X71+X72+X73=1;
X81+X82+X83=1;
X91+X92+X93=1;
X101+X102+X103=1;
X111+X112+X113=1;
X121+X122+X123=1;


Y11+Y12+Y13=1;
Y21+Y22+Y23=1;
Y31+Y32+Y33=1;
Y41+Y42+Y43=1;
Y51+Y52+Y53=1;
Y61+Y62+Y63=1;
Y71+Y72+Y73=1;
Y81+Y82+Y83=1;
Y91+Y92+Y93=1;
Y101+Y102+Y103=1;
Y111+Y112+Y113=1;


@GIN(X11);
@GIN(X12);
@GIN(X13);
@GIN(X21);
@GIN(X22);
@GIN(X23);
@GIN(X31);
@GIN(X32);
@GIN(X33);
@GIN(X41);
@GIN(X42);
@GIN(X43);
@GIN(X51);
@GIN(X52);
@GIN(X53);
@GIN(X61);
@GIN(X62);
@GIN(X63);
@GIN(X71);
@GIN(X72);
@GIN(X73);
@GIN(X81);
@GIN(X82);
@GIN(X83);
@GIN(X91);
@GIN(X92);
@GIN(X93);
@GIN(X101);
@GIN(X102);
@GIN(X103);
@GIN(X111);
@GIN(X112);
@GIN(X113);
@GIN(X121);
@GIN(X122);
@GIN(X123);

@GIN(Y11);
@GIN(Y12);
@GIN(Y13);
@GIN(Y21);
@GIN(Y22);
@GIN(Y23);
@GIN(Y31);
@GIN(Y32);
@GIN(Y33);
@GIN(Y41);
@GIN(Y42);
@GIN(Y43);
@GIN(Y51);
@GIN(Y52);
@GIN(Y53);
@GIN(Y61);
@GIN(Y62);
@GIN(Y63);
@GIN(Y71);
@GIN(Y72);
@GIN(Y73);
@GIN(Y81);
@GIN(Y82);
@GIN(Y83);
@GIN(Y91);
@GIN(Y92);
@GIN(Y93);
@GIN(Y101);
@GIN(Y102);
@GIN(Y103);
@GIN(Y111);
@GIN(Y112);
@GIN(Y113);
END
```

APPENDIX F. VIRTUAL CELL OBTAINED BY USING KO'S METHOD

The Ko's virtual cell formation procedure is employed in the experiment. The generated virtual cells for each production session are as follows:

EXAMPLE 1 Production Session 1

There are 11 Jobs in the file
Job[1](size= 8,demand= 100) : 1 4 7 4 8 9
Job[2](size= 6,demand= 120) : 1 2 4 8
Job[3](size= 5,demand= 200) : 4 7 9
Job[4](size= 6,demand= 50) : 4 7 4 8
Job[5](size= 9,demand= 90) : 3 5 2 6 4 8 9
Job[6](size= 6,demand= 80) : 3 5 7 8
Job[7](size= 6,demand= 70) : 2 6 3 5
Job[8](size= 7,demand= 75) : 3 5 4 7 8
Job[9](size= 5,demand= 70) : 11 7 12
Job[10](size= 4,demand= 50) : 11 12
Job[11](size= 7,demand= 70) : 11 7 10 11 12

Virtual_Cell[1] (size= 4, demand= 715): 4 7 8 9
Virtual_Cell[2] (size= 4, demand= 190): 11 7 12 10
Virtual_Cell[3] (size= 2, demand= 220): 1 2
Virtual_Cell[4] (size= 2, demand= 315): 3 5
Virtual_Cell[5] (size= 2, demand= 160): 2 6

Job[1] (size= 2, demand= 100): C3 C1
Job[2] (size= 2, demand= 120): C3 C1
Job[3] (size= 1, demand= 200): C1
Job[4] (size= 1, demand= 50): C1
Job[5] (size= 3, demand= 90): C4 C5 C1
Job[6] (size= 2, demand= 80): C4 C1
Job[7] (size= 2, demand= 70): C5 C4
Job[8] (size= 2, demand= 75): C4 C1
Job[9] (size= 1, demand= 70): C2
Job[10] (size= 1, demand= 50): C2
Job[11] (size= 1, demand= 70): C2

EXAMPLE 1 Production Session 2

There are 11 Jobs in the file
Job[1](size= 6,demand= 100) : 7 8 7 9
Job[2](size= 8,demand= 120) : 1 2 4 7 8 9
Job[3](size= 5,demand= 200) : 4 8 9
Job[4](size= 6,demand= 50) : 4 7 8 9
Job[5](size= 4,demand= 60) : 7 8
Job[6](size= 7,demand= 90) : 3 5 4 8 9
Job[7](size= 8,demand= 80) : 2 3 4 5 8 9
Job[8](size= 8,demand= 75) : 3 5 6 4 8 9
Job[9](size= 6,demand= 70) : 6 8 10 12
Job[10](size= 7,demand= 50) : 3 6 8 10 12
Job[11](size= 6,demand= 70) : 3 8 10 12

Virtual_Cell[1] (size= 2, demand= 70): 3 8
Virtual_Cell[2] (size= 4, demand= 245): 4 5 2 3
Virtual_Cell[3] (size= 4, demand= 775): 4 8 7 9
Virtual_Cell[4] (size= 3, demand= 190): 8 10 12
Virtual_Cell[5] (size= 2, demand= 195): 3 6
Virtual_Cell[6] (size= 3, demand= 120): 2 4 1

Job[1] (size= 1, demand= 100): C3
Job[2] (size= 2, demand= 120): C6 C3
Job[3] (size= 1, demand= 200): C3
Job[4] (size= 1, demand= 50): C3
Job[5] (size= 1, demand= 60): C3
Job[6] (size= 2, demand= 90): C2 C3
Job[7] (size= 2, demand= 80): C2 C3
Job[8] (size= 3, demand= 75): C2 C5 C3
Job[9] (size= 2, demand= 70): C5 C4
Job[10] (size= 2, demand= 50): C5 C4
Job[11] (size= 2, demand= 70): C1 C4

EXAMPLE 1 Production Session 3

There are 11 Jobs in the file
Job[1](size= 5,demand= 90) : 4 8 9
Job[2](size= 6,demand= 100) : 4 8 7 9
Job[3](size= 7,demand= 120) : 4 7 8 7 9
Job[4](size= 6,demand= 200) : 4 7 8 9
Job[5](size= 7,demand= 75) : 11 12 8 9 7
Job[6](size= 6,demand= 90) : 1 7 8 9
Job[7](size= 6,demand= 70) : 1 7 9 11
Job[8](size= 6,demand= 50) : 1 7 11 12
Job[9](size= 5,demand= 70) : 10 11 12
Job[10](size= 7,demand= 90) : 11 7 12 1 7
Job[11](size= 7,demand= 100) : 1 7 11 12 11

Virtual_Cell[1] (size= 4, demand= 745): 4 7 8 9
Virtual_Cell[2] (size= 2, demand= 70): 10 11
Virtual_Cell[3] (size= 4, demand= 765): 11 7 12 1

Job[1] (size= 1, demand= 90): C1
Job[2] (size= 1, demand= 100): C1
Job[3] (size= 1, demand= 120): C1
Job[4] (size= 1, demand= 200): C1
Job[5] (size= 2, demand= 75): C3 C1
Job[6] (size= 2, demand= 90): C3 C1
Job[7] (size= 3, demand= 70): C3 C1 C3
Job[8] (size= 1, demand= 50): C3
Job[9] (size= 2, demand= 70): C2 C3
Job[10] (size= 1, demand= 90): C3
Job[11] (size= 1, demand= 100): C3

EXAMPLE 1 Production Session 4

There are 14 Jobs in the file
Job[1](size= 5,demand= 100) : 6 7 8
Job[2](size= 5,demand= 120) : 6 8 7
Job[3](size= 8,demand= 150) : 6 8 7 8 7 9
Job[4](size= 5,demand= 80) : 6 7 9
Job[5](size= 8,demand= 75) : 3 5 7 9 8 9
Job[6](size= 7,demand= 150) : 3 5 4 7 8
Job[7](size= 8,demand= 170) : 3 4 5 4 8 9
Job[8](size= 7,demand= 90) : 3 5 4 7 8
Job[9](size= 6,demand= 80) : 3 4 3 5

Job[10](size= 6,demand= 190) : 3 4 7 8
Job[11](size= 6,demand= 110) : 1 7 11 12
Job[12](size= 5,demand= 60) : 11 7 12
Job[13](size= 5,demand= 80) : 10 11 12
Job[14](size= 6,demand= 90) : 12 11 7 11

Virtual_Cell[1] (size= 2, demand= 110): 1 7
Virtual_Cell[2] (size= 4, demand= 755): 3 5 7 4
Virtual_Cell[3] (size= 4, demand= 1125): 6 7 8 9
Virtual_Cell[4] (size= 2, demand= 80): 10 11
Virtual_Cell[5] (size= 3, demand= 340): 7 12 11

Job[1] (size= 1, demand= 100): C3
Job[2] (size= 1, demand= 120): C3
Job[3] (size= 1, demand= 150): C3
Job[4] (size= 1, demand= 80): C3
Job[5] (size= 2, demand= 75): C2 C3
Job[6] (size= 2, demand= 150): C2 C3
Job[7] (size= 2, demand= 170): C2 C3
Job[8] (size= 2, demand= 90): C2 C3
Job[9] (size= 1, demand= 80): C2
Job[10] (size= 2, demand= 190): C2 C3
Job[11] (size= 2, demand= 110): C1 C5
Job[12] (size= 1, demand= 60): C5
Job[13] (size= 2, demand= 80): C4 C5
Job[14] (size= 1, demand= 90): C5

EXAMPLE 1 Production Session 5

There are 14 Jobs in the file
Job[1](size= 4,demand= 100) : 1 6 4 7
Job[2](size= 4,demand= 150) : 1 6 11 12
Job[3](size= 3,demand= 120) : 1 7 9
Job[4](size= 6,demand= 80) : 1 6 7 1 11 12
Job[5](size= 4,demand= 75) : 1 6 7 9
Job[6](size= 3,demand= 150) : 7 11 12
Job[7](size= 4,demand= 90) : 3 5 4 9
Job[8](size= 4,demand= 80) : 3 5 7 8
Job[9](size= 5,demand= 190) : 3 5 4 7 8
Job[10](size= 5,demand= 110) : 3 5 4 8 9
Job[11](size= 5,demand= 75) : 4 7 8 7 9
Job[12](size= 3,demand= 100) : 4 7 9
Job[13](size= 4,demand= 110) : 3 4 8 9
Job[14](size= 6,demand= 170) : 3 4 3 5 8 9

Virtual_Cell[1] (size= 2, demand= 405): 1 6
Virtual_Cell[2] (size= 2, demand= 350): 1 7
Virtual_Cell[3] (size= 3, demand= 750): 3 5 4
Virtual_Cell[4] (size= 4, demand= 1220): 4 7 8 9
Virtual_Cell[5] (size= 2, demand= 380): 11 12

Job[1] (size= 2, demand= 100): C1 C4
Job[2] (size= 2, demand= 150): C1 C5
Job[3] (size= 2, demand= 120): C2 C4
Job[4] (size= 3, demand= 80): C1 C2 C5
Job[5] (size= 2, demand= 75): C1 C4
Job[6] (size= 2, demand= 150): C2 C5
Job[7] (size= 2, demand= 90): C3 C4
Job[8] (size= 2, demand= 80): C3 C4
Job[9] (size= 2, demand= 190): C3 C4
Job[10] (size= 2, demand= 110): C3 C4
Job[11] (size= 1, demand= 75): C4
Job[12] (size= 1, demand= 100): C4
Job[13] (size= 2, demand= 110): C3 C4
Job[14] (size= 2, demand= 170): C3 C4

EXAMPLE 1 Production Session 6

There are 8 Jobs in the file
Job[1](size= 7,demand= 100) : 1 6 10 7 9
Job[2](size= 7,demand= 130) : 6 10 7 8 9
Job[3](size= 5,demand= 110) : 6 7 10
Job[4](size= 6,demand= 90) : 1 6 7 10
Job[5](size= 6,demand= 90) : 3 5 7 8
Job[6](size= 8,demand= 80) : 3 5 7 9 8 9
Job[7](size= 5,demand= 190) : 10 11 12
Job[8](size= 5,demand= 200) : 7 11 12

Virtual_Cell[1] (size= 3, demand= 600): 7 8 9
Virtual_Cell[2] (size= 2, demand= 390): 11 12
Virtual_Cell[3] (size= 2, demand= 170): 3 5
Virtual_Cell[4] (size= 4, demand= 620): 1 6 7 10

Job[1] (size= 2, demand= 100): C4 C1
Job[2] (size= 2, demand= 130): C4 C1
Job[3] (size= 1, demand= 110): C4
Job[4] (size= 1, demand= 90): C4
Job[5] (size= 2, demand= 90): C3 C1
Job[6] (size= 2, demand= 80): C3 C1
Job[7] (size= 2, demand= 190): C4 C2
Job[8] (size= 2, demand= 200): C1 C2

EXAMPLE 1 Production Session 7

There are 12 Jobs in the file
Job[1](size= 6,demand= 120) : 1 4 8 9
Job[2](size= 6,demand= 110) : 7 8 7 9
Job[3](size= 6,demand= 105) : 4 7 4 8
Job[4](size= 5,demand= 90) : 4 7 9
Job[5](size= 7,demand= 70) : 1 2 1 6 10
Job[6](size= 6,demand= 80) : 1 6 7 10
Job[7](size= 7,demand= 60) : 1 2 6 7 10
Job[8](size= 6,demand= 50) : 7 12 11 12
Job[9](size= 6,demand= 45) : 12 11 7 11
Job[10](size= 5,demand= 50) : 11 7 10
Job[11](size= 4,demand= 60) : 11 12
Job[12](size= 6,demand= 70) : 1 7 11 12

Virtual_Cell[1] (size= 4, demand= 210): 1 2 6 10
Virtual_Cell[2] (size= 5, demand= 625): 1 4 8 9 7
Virtual_Cell[3] (size= 2, demand= 190): 7 10
Virtual_Cell[4] (size= 4, demand= 275): 7 12 11 1

Job[1] (size= 1, demand= 120): C2
Job[2] (size= 1, demand= 110): C2
Job[3] (size= 1, demand= 105): C2
Job[4] (size= 1, demand= 90): C2
Job[5] (size= 1, demand= 70): C1
Job[6] (size= 2, demand= 80): C1 C3
Job[7] (size= 2, demand= 60): C1 C3
Job[8] (size= 1, demand= 50): C4
Job[9] (size= 1, demand= 45): C4
Job[10] (size= 2, demand= 50): C4 C3
Job[11] (size= 1, demand= 60): C4
Job[12] (size= 1, demand= 70): C4

EXAMPLE 1 Production Session 8

There are 10 Jobs in the file

Job[1](size= 5,demand= 100) : 4 7 11
Job[2](size= 7,demand= 70) : 4 7 9 12 11
Job[3](size= 5,demand= 70) : 1 6 7
Job[4](size= 6,demand= 80) : 1 6 7 9
Job[5](size= 7,demand= 80) : 5 3 4 8 9
Job[6](size= 8,demand= 90) : 3 4 5 4 8 7
Job[7](size= 7,demand= 40) : 3 5 3 5 4
Job[8](size= 8,demand= 200) : 3 4 8 9 7 8
Job[9](size= 6,demand= 100) : 1 6 7 10
Job[10](size= 7,demand= 200) : 1 6 9 7 8

Virtual_Cell[1] (size= 2, demand= 450): 1 6
Virtual_Cell[2] (size= 6, demand= 1290): 4 7 8 11 9 12
Virtual_Cell[3] (size= 2, demand= 100): 7 10
Virtual_Cell[4] (size= 3, demand= 410): 3 4 5

Job[1] (size= 1, demand= 100): C2
Job[2] (size= 1, demand= 70): C2
Job[3] (size= 2, demand= 70): C1 C2
Job[4] (size= 2, demand= 80): C1 C2
Job[5] (size= 2, demand= 80): C4 C2
Job[6] (size= 2, demand= 90): C4 C2
Job[7] (size= 1, demand= 40): C4
Job[8] (size= 2, demand= 200): C4 C2
Job[9] (size= 2, demand= 100): C1 C3
Job[10] (size= 2, demand= 200): C1 C2

EXAMPLE 1 Production Session 9

There are 17 Jobs in the file
Job[1](size= 6,demand= 200) : 1 4 8 9
Job[2](size= 8,demand= 300) : 1 4 7 4 7 9
Job[3](size= 8,demand= 100) : 1 2 4 7 8 9
Job[4](size= 7,demand= 300) : 4 9 4 7 8
Job[5](size= 7,demand= 200) : 1 6 10 7 9
Job[6](size= 7,demand= 100) : 6 10 7 8 9
Job[7](size= 6,demand= 200) : 4 8 7 9
Job[8](size= 5,demand= 100) : 11 7 12
Job[9](size= 5,demand= 300) : 6 7 10
Job[10](size= 6,demand= 75) : 1 5 7 8
Job[11](size= 4,demand= 50) : 1 7
Job[12](size= 7,demand= 80) : 8 9 1 6 10
Job[13](size= 7,demand= 300) : 1 2 6 7 8
Job[14](size= 7,demand= 250) : 4 8 9 1 5
Job[15](size= 7,demand= 210) : 4 7 8 7 9
Job[16](size= 7,demand= 140) : 11 7 12 1 7
Job[17](size= 7,demand= 270) : 1 5 4 7 8

Virtual_Cell[1] (size= 2, demand= 400): 1 2
Virtual_Cell[2] (size= 2, demand= 1095): 1 5
Virtual_Cell[3] (size= 4, demand= 290): 1 7 11 12
Virtual_Cell[4] (size= 3, demand= 680): 1 6 10
Virtual_Cell[5] (size= 4, demand= 2210): 4 8 9 7
Virtual_Cell[6] (size= 2, demand= 300): 7 10
Virtual_Cell[7] (size= 3, demand= 375): 6 7 8

Job[1] (size= 2, demand= 200): C2 C5
Job[2] (size= 2, demand= 300): C2 C5
Job[3] (size= 2, demand= 100): C1 C5
Job[4] (size= 1, demand= 300): C5
Job[5] (size= 2, demand= 200): C4 C5
Job[6] (size= 2, demand= 100): C4 C5
Job[7] (size= 1, demand= 200): C5
Job[8] (size= 1, demand= 100): C3
Job[9] (size= 2, demand= 300): C4 C6
Job[10] (size= 2, demand= 75): C2 C7
Job[11] (size= 1, demand= 50): C3
Job[12] (size= 2, demand= 80): C5 C4
Job[13] (size= 2, demand= 300): C1 C7
Job[14] (size= 2, demand= 250): C5 C2
Job[15] (size= 1, demand= 210): C5
Job[16] (size= 1, demand= 140): C3
Job[17] (size= 2, demand= 270): C2 C5

EXAMPLE 1 Production Session 10

There are 16 Jobs in the file
Job[1](size= 6,demand= 200) : 4 7 8 9
Job[2](size= 9,demand= 100) : 3 5 4 7 4 8 9
Job[3](size= 7,demand= 100) : 3 5 4 8 9
Job[4](size= 7,demand= 200) : 4 7 4 8 7
Job[5](size= 5,demand= 100) : 11 7 12
Job[6](size= 4,demand= 100) : 11 12
Job[7](size= 5,demand= 300) : 11 7 10
Job[8](size= 8,demand= 300) : 1 7 11 10 11 12
Job[9](size= 7,demand= 45) : 3 5 1 6 10
Job[10](size= 7,demand= 80) : 8 9 1 6 10
Job[11](size= 7,demand= 300) : 1 2 6 1 6
Job[12](size= 6,demand= 110) : 1 7 3 5
Job[13](size= 6,demand= 225) : 1 6 7 10
Job[14](size= 5,demand= 200) : 1 2 6
Job[15](size= 6,demand= 150) : 3 5 6 10
Job[16](size= 7,demand= 140) : 11 7 12 1 7

Virtual_Cell[1] (size= 5, demand= 1275): 1 7 11 10 12
Virtual_Cell[2] (size= 3, demand= 1000): 1 6 10
Virtual_Cell[3] (size= 2, demand= 505): 3 5
Virtual_Cell[4] (size= 4, demand= 680): 8 9 4 7
Virtual_Cell[5] (size= 2, demand= 500): 1 2

Job[1] (size= 1, demand= 200): C4
Job[2] (size= 2, demand= 100): C3 C4
Job[3] (size= 2, demand= 100): C3 C4
Job[4] (size= 1, demand= 200): C4
Job[5] (size= 1, demand= 100): C1
Job[6] (size= 1, demand= 100): C1
Job[7] (size= 1, demand= 300): C1
Job[8] (size= 1, demand= 300): C1
Job[9] (size= 2, demand= 45): C3 C2
Job[10] (size= 2, demand= 80): C4 C2
Job[11] (size= 2, demand= 300): C5 C2
Job[12] (size= 2, demand= 110): C1 C3
Job[13] (size= 2, demand= 225): C2 C1
Job[14] (size= 2, demand= 200): C5 C2
Job[15] (size= 2, demand= 150): C3 C2
Job[16] (size= 1, demand= 140): C1

EXAMPLE 2 Production Session 1

There are 11 Jobs in the file
Job[1](size= 6,demand= 472) : 4 6 5 8
Job[2](size= 6,demand= 182) : 2 3 7 6
Job[3](size= 4,demand= 52) : 1 5
Job[4](size= 4,demand= 71) : 4 2
Job[5](size= 4,demand= 180) : 8 3
Job[6](size= 4,demand= 81) : 8 4
Job[7](size= 5,demand= 272) : 5 2 4
Job[8](size= 5,demand= 310) : 8 6 1
Job[9](size= 6,demand= 459) : 1 6 4 6
Job[10](size= 5,demand= 72) : 4 2 7

Job[11](size= 5,demand= 142) : 4 1 4

Virtual_Cell[1] (size= 6, demand= 1931): 1 4 6 5 8 2
Virtual_Cell[2] (size= 2, demand= 182): 2 3
Virtual_Cell[3] (size= 2, demand= 180): 8 3
Virtual_Cell[4] (size= 2, demand= 254): 7 6

Job[1] (size= 1, demand= 472): C1
Job[2] (size= 2, demand= 182): C2 C4
Job[3] (size= 1, demand= 52): C1
Job[4] (size= 1, demand= 71): C1
Job[5] (size= 1, demand= 180): C3
Job[6] (size= 1, demand= 81): C1
Job[7] (size= 1, demand= 272): C1
Job[8] (size= 1, demand= 310): C1
Job[9] (size= 1, demand= 459): C1
Job[10] (size= 2, demand= 72): C1 C4
Job[11] (size= 1, demand= 142): C1

EXAMPLE 2 Production Session 2

There are 12 Jobs in the file
Job[1](size= 5,demand= 279) : 4 2 6
Job[2](size= 4,demand= 14) : 8 7
Job[3](size= 5,demand= 187) : 6 7 5
Job[4](size= 6,demand= 39) : 2 4 7 1
Job[5](size= 6,demand= 11) : 1 5 2 4
Job[6](size= 7,demand= 305) : 4 6 7 8 2
Job[7](size= 6,demand= 55) : 5 1 7 1
Job[8](size= 7,demand= 337) : 5 2 6 8 3
Job[9](size= 6,demand= 404) : 6 7 5 7
Job[10](size= 5,demand= 180) : 6 8 2
Job[11](size= 5,demand= 32) : 6 7 2
Job[12](size= 4,demand= 111) : 7 6

Virtual_Cell[1] (size= 5, demand= 1344): 4 2 5 7 1
Virtual_Cell[2] (size= 2, demand= 305): 4 6
Virtual_Cell[3] (size= 2, demand= 485): 8 2
Virtual_Cell[4] (size= 2, demand= 337): 8 3
Virtual_Cell[5] (size= 2, demand= 1835): 7 6
Virtual_Cell[6] (size= 2, demand= 14): 8 7

Job[1] (size= 2, demand= 279): C1 C5
Job[2] (size= 1, demand= 14): C6
Job[3] (size= 2, demand= 187): C5 C1
Job[4] (size= 1, demand= 39): C1
Job[5] (size= 1, demand= 11): C1
Job[6] (size= 3, demand= 305): C2 C5 C3
Job[7] (size= 1, demand= 55): C1
Job[8] (size= 3, demand= 337): C1 C5 C4
Job[9] (size= 2, demand= 404): C5 C1
Job[10] (size= 2, demand= 180): C5 C3
Job[11] (size= 2, demand= 32): C5 C1
Job[12] (size= 1, demand= 111): C5

EXAMPLE 2 Production Session 3

There are 8 Jobs in the file
Job[1](size= 5,demand= 97) : 3 7 4
Job[2](size= 4,demand= 71) : 1 6
Job[3](size= 4,demand= 227) : 1 3
Job[4](size= 5,demand= 330) : 2 1 8
Job[5](size= 5,demand= 205) : 2 8 6
Job[6](size= 5,demand= 36) : 6 2 4
Job[7](size= 5,demand= 355) : 3 1 3
Job[8](size= 6,demand= 341) : 7 5 8 4

Virtual_Cell[1] (size= 4, demand= 312): 1 6 2 4
Virtual_Cell[2] (size= 2, demand= 330): 1 8
Virtual_Cell[3] (size= 2, demand= 535): 2 8
Virtual_Cell[4] (size= 2, demand= 438): 7 4
Virtual_Cell[5] (size= 3, demand= 341): 7 5 8
Virtual_Cell[6] (size= 2, demand= 679): 1 3

Job[1] (size= 2, demand= 97): C6 C4
Job[2] (size= 1, demand= 71): C1
Job[3] (size= 1, demand= 227): C6
Job[4] (size= 2, demand= 330): C3 C2
Job[5] (size= 2, demand= 205): C3 C1
Job[6] (size= 1, demand= 36): C1
Job[7] (size= 1, demand= 355): C6
Job[8] (size= 2, demand= 341): C5 C4

EXAMPLE 2 Production Session 4

There are 13 Jobs in the file
Job[1](size= 4,demand= 135) : 4 5
Job[2](size= 5,demand= 304) : 4 2 6
Job[3](size= 6,demand= 478) : 3 5 2 1
Job[4](size= 7,demand= 290) : 6 3 4 1 5
Job[5](size= 6,demand= 29) : 2 6 1 3
Job[6](size= 7,demand= 335) : 5 4 3 5 4
Job[7](size= 5,demand= 276) : 6 4 6
Job[8](size= 7,demand= 287) : 1 6 8 4 1
Job[9](size= 7,demand= 395) : 2 6 4 1 4
Job[10](size= 5,demand= 462) : 1 6 2
Job[11](size= 6,demand= 182) : 4 5 4 3
Job[12](size= 4,demand= 334) : 7 1
Job[13](size= 4,demand= 225) : 4 3

Virtual_Cell[1] (size= 4, demand= 1955): 6 2 1 8
Virtual_Cell[2] (size= 3, demand= 319): 6 3 1
Virtual_Cell[3] (size= 2, demand= 334): 7 1
Virtual_Cell[4] (size= 4, demand= 2631): 1 5 4 3
Virtual_Cell[5] (size= 2, demand= 276): 4 6

Job[1] (size= 1, demand= 135): C4
Job[2] (size= 2, demand= 304): C4 C1
Job[3] (size= 2, demand= 478): C4 C1
Job[4] (size= 2, demand= 290): C2 C4
Job[5] (size= 2, demand= 29): C1 C2
Job[6] (size= 1, demand= 335): C4
Job[7] (size= 1, demand= 276): C5
Job[8] (size= 2, demand= 287): C1 C4
Job[9] (size= 2, demand= 395): C1 C4
Job[10] (size= 1, demand= 462): C1
Job[11] (size= 1, demand= 182): C4
Job[12] (size= 1, demand= 334): C3
Job[13] (size= 1, demand= 225): C4

EXAMPLE 2 Production Session 5

There are 10 Jobs in the file
Job[1](size= 5,demand= 304) : 8 2 4
Job[2](size= 4,demand= 336) : 2 7
Job[3](size= 4,demand= 410) : 7 6
Job[4](size= 5,demand= 11) : 6 7 1
Job[5](size= 5,demand= 300) : 5 2 6

Job[6](size= 4,demand= 172) : 7 3
Job[7](size= 5,demand= 185) : 2 4 5
Job[8](size= 4,demand= 260) : 6 5
Job[9](size= 4,demand= 108) : 8 6
Job[10](size= 5,demand= 75) : 4 1 5

Virtual_Cell[1] (size= 6, demand= 1773): 2 6 7 5 4 1
Virtual_Cell[2] (size= 2, demand= 172): 7 3
Virtual_Cell[3] (size= 4, demand= 412): 8 6 2 4

Job[1] (size= 1, demand= 304): C3
Job[2] (size= 1, demand= 336): C1
Job[3] (size= 1, demand= 410): C1
Job[4] (size= 1, demand= 11): C1
Job[5] (size= 1, demand= 300): C1
Job[6] (size= 1, demand= 172): C2
Job[7] (size= 1, demand= 185): C1
Job[8] (size= 1, demand= 260): C1
Job[9] (size= 1, demand= 108): C3
Job[10] (size= 1, demand= 75): C1

EXAMPLE 2 Production Session 6

There are 7 Jobs in the file
Job[1](size= 6,demand= 84) : 8 1 4 8
Job[2](size= 4,demand= 305) : 8 5
Job[3](size= 4,demand= 118) : 3 8
Job[4](size= 6,demand= 85) : 8 5 7 8
Job[5](size= 6,demand= 72) : 3 2 4 5
Job[6](size= 5,demand= 59) : 3 5 6
Job[7](size= 5,demand= 48) : 1 6 2

Virtual_Cell[1] (size= 6, demand= 179): 1 6 3 2 4 5
Virtual_Cell[2] (size= 5, demand= 651): 3 5 8 1 4
Virtual_Cell[3] (size= 2, demand= 85): 7 8

Job[1] (size= 1, demand= 84): C2
Job[2] (size= 1, demand= 305): C2
Job[3] (size= 1, demand= 118): C2
Job[4] (size= 2, demand= 85): C2 C3
Job[5] (size= 1, demand= 72): C1
Job[6] (size= 2, demand= 59): C2 C1
Job[7] (size= 1, demand= 48): C1

EXAMPLE 2 Production Session 7

There are 15 Jobs in the file
Job[1](size= 5,demand= 444) : 3 8 6
Job[2](size= 7,demand= 101) : 8 4 8 6 3
Job[3](size= 5,demand= 302) : 5 6 3
Job[4](size= 5,demand= 74) : 2 7 8
Job[5](size= 5,demand= 72) : 2 8 3
Job[6](size= 6,demand= 127) : 8 1 4 3
Job[7](size= 7,demand= 380) : 5 8 2 7 4
Job[8](size= 4,demand= 343) : 1 7
Job[9](size= 5,demand= 388) : 1 4 2
Job[10](size= 4,demand= 254) : 4 1
Job[11](size= 6,demand= 166) : 1 2 7 3
Job[12](size= 7,demand= 162) : 6 1 2 8 4
Job[13](size= 4,demand= 116) : 8 4
Job[14](size= 7,demand= 21) : 8 4 3 6 1
Job[15](size= 5,demand= 124) : 3 5 1

Virtual_Cell[1] (size= 2, demand= 1477): 4 1
Virtual_Cell[2] (size= 2, demand= 21): 4 3
Virtual_Cell[3] (size= 3, demand= 217): 4 8 6
Virtual_Cell[4] (size= 3, demand= 1053): 5 1 6
Virtual_Cell[5] (size= 2, demand= 380): 5 8
Virtual_Cell[6] (size= 2, demand= 343): 1 7
Virtual_Cell[7] (size= 2, demand= 1008): 2 7
Virtual_Cell[8] (size= 2, demand= 234): 2 8
Virtual_Cell[9] (size= 2, demand= 1558): 3 8

Job[1] (size= 2, demand= 444): C9 C4
Job[2] (size= 2, demand= 101): C3 C9
Job[3] (size= 2, demand= 302): C4 C9
Job[4] (size= 2, demand= 74): C7 C9
Job[5] (size= 2, demand= 72): C8 C9
Job[6] (size= 3, demand= 127): C9 C1 C9
Job[7] (size= 3, demand= 380): C5 C7 C1
Job[8] (size= 1, demand= 343): C6
Job[9] (size= 2, demand= 388): C1 C7
Job[10] (size= 1, demand= 254): C1
Job[11] (size= 3, demand= 166): C1 C7 C9
Job[12] (size= 3, demand= 162): C4 C8 C1
Job[13] (size= 1, demand= 116): C3
Job[14] (size= 3, demand= 21): C9 C2 C4
Job[15] (size= 2, demand= 124): C9 C4

EXAMPLE 2 Production Session 8

There are 9 Jobs in the file
Job[1](size= 6,demand= 303) : 7 8 1 5
Job[2](size= 5,demand= 5) : 7 5 3
Job[3](size= 4,demand= 3) : 4 7
Job[4](size= 6,demand= 456) : 3 7 4 7
Job[5](size= 5,demand= 249) : 3 4 1
Job[6](size= 5,demand= 354) : 1 3 8
Job[7](size= 4,demand= 32) : 7 1
Job[8](size= 5,demand= 289) : 8 5 2
Job[9](size= 4,demand= 4) : 2 4

Virtual_Cell[1] (size= 6, demand= 2050): 3 4 7 8 1 5
Virtual_Cell[2] (size= 2, demand= 293): 2 4

Job[1] (size= 1, demand= 303): C1
Job[2] (size= 1, demand= 5): C1
Job[3] (size= 1, demand= 3): C1
Job[4] (size= 1, demand= 456): C1
Job[5] (size= 1, demand= 249): C1
Job[6] (size= 1, demand= 354): C1
Job[7] (size= 1, demand= 32): C1
Job[8] (size= 2, demand= 289): C1 C2
Job[9] (size= 1, demand= 4): C2

EXAMPLE 2 Production Session 9

There are 11 Jobs in the file
Job[1](size= 4,demand= 134) : 4 8
Job[2](size= 6,demand= 262) : 8 3 5 2
Job[3](size= 4,demand= 374) : 8 2
Job[4](size= 5,demand= 376) : 5 3 2
Job[5](size= 5,demand= 456) : 3 4 7
Job[6](size= 5,demand= 254) : 6 8 2
Job[7](size= 5,demand= 167) : 5 7 5
Job[8](size= 5,demand= 279) : 5 7 6
Job[9](size= 6,demand= 95) : 5 8 5 6
Job[10](size= 4,demand= 1) : 5 6

Job[11](size= 6,demand= 401) : 5 7 8 6

Virtual_Cell[1] (size= 4, demand= 1817): 3 2 8 5
Virtual_Cell[2] (size= 2, demand= 456): 4 7
Virtual_Cell[3] (size= 3, demand= 1030): 5 6 8
Virtual_Cell[4] (size= 2, demand= 1223): 5 7
Virtual_Cell[5] (size= 2, demand= 134): 4 8

Job[1] (size= 1, demand= 134): C5
Job[2] (size= 1, demand= 262): C1
Job[3] (size= 1, demand= 374): C1
Job[4] (size= 2, demand= 376): C4 C1
Job[5] (size= 2, demand= 456): C1 C2
Job[6] (size= 2, demand= 254): C3 C1
Job[7] (size= 1, demand= 167): C4
Job[8] (size= 2, demand= 279): C4 C3
Job[9] (size= 2, demand= 95): C1 C3
Job[10] (size= 1, demand= 1): C3
Job[11] (size= 2, demand= 401): C4 C3

EXAMPLE 2 Production Session 10

There are 15 Jobs in the file
Job[1](size= 7,demand= 158) : 5 3 1 3 4
Job[2](size= 5,demand= 371) : 5 6 2
Job[3](size= 6,demand= 197) : 1 8 1 8
Job[4](size= 4,demand= 443) : 3 5
Job[5](size= 4,demand= 215) : 5 1
Job[6](size= 7,demand= 186) : 5 3 2 6 1
Job[7](size= 7,demand= 199) : 7 2 6 3 1
Job[8](size= 5,demand= 459) : 5 1 2
Job[9](size= 5,demand= 373) : 4 6 8
Job[10](size= 6,demand= 191) : 2 4 2 7
Job[11](size= 5,demand= 318) : 3 6 1
Job[12](size= 6,demand= 108) : 5 4 7 8
Job[13](size= 5,demand= 334) : 1 3 4
Job[14](size= 6,demand= 277) : 3 4 7 1
Job[15](size= 7,demand= 436) : 8 6 1 4 8

Virtual_Cell[1] (size= 5, demand= 2683): 3 1 5 6 2
Virtual_Cell[2] (size= 3, demand= 1978): 3 2 4
Virtual_Cell[3] (size= 4, demand= 2094): 6 1 8 7
Virtual_Cell[4] (size= 2, demand= 191): 7 2
Virtual_Cell[5] (size= 2, demand= 108): 5 4
Virtual_Cell[6] (size= 2, demand= 436): 4 8

Job[1] (size= 2, demand= 158): C1 C2
Job[2] (size= 1, demand= 371): C1
Job[3] (size= 1, demand= 197): C3
Job[4] (size= 1, demand= 443): C1
Job[5] (size= 1, demand= 215): C1
Job[6] (size= 3, demand= 186): C1 C2 C3
Job[7] (size= 2, demand= 199): C3 C1
Job[8] (size= 2, demand= 459): C1 C2
Job[9] (size= 2, demand= 373): C2 C3
Job[10] (size= 2, demand= 191): C2 C4
Job[11] (size= 2, demand= 318): C1 C3
Job[12] (size= 2, demand= 108): C5 C3
Job[13] (size= 2, demand= 334): C1 C2
Job[14] (size= 2, demand= 277): C2 C3
Job[15] (size= 2, demand= 436): C3 C6

EXAMPLE 3 Production Session 1

There are 7 Jobs in the file
Job[1](size= 5,demand= 378) : 2 3 2
Job[2](size= 6,demand= 432) : 5 1 5 4
Job[3](size= 5,demand= 266) : 5 3 2
Job[4](size= 4,demand= 481) : 3 5
Job[5](size= 7,demand= 371) : 2 5 2 3 1
Job[6](size= 6,demand= 417) : 1 2 3 4
Job[7](size= 4,demand= 471) : 5 4

Virtual_Cell[1] (size= 3, demand= 1913): 3 2 5
Virtual_Cell[2] (size= 2, demand= 1320): 5 4
Virtual_Cell[3] (size= 2, demand= 1220): 5 1

Job[1] (size= 1, demand= 378): C1
Job[2] (size= 2, demand= 432): C3 C2
Job[3] (size= 1, demand= 266): C1
Job[4] (size= 1, demand= 481): C1
Job[5] (size= 2, demand= 371): C1 C3
Job[6] (size= 3, demand= 417): C3 C1 C2
Job[7] (size= 1, demand= 471): C2

EXAMPLE 3 Production Session 2

There are 10 Jobs in the file
Job[1](size= 5,demand= 138) : 2 4 1
Job[2](size= 4,demand= 188) : 5 2
Job[3](size= 6,demand= 287) : 4 1 2 5
Job[4](size= 4,demand= 57) : 3 1
Job[5](size= 4,demand= 442) : 3 2
Job[6](size= 4,demand= 460) : 3 2
Job[7](size= 4,demand= 82) : 1 4
Job[8](size= 6,demand= 206) : 4 3 1 2
Job[9](size= 5,demand= 349) : 5 2 5
Job[10](size= 4,demand= 197) : 2 4

Virtual_Cell[1] (size= 4, demand= 1869): 3 2 1 4
Virtual_Cell[2] (size= 2, demand= 824): 2 5

Job[1] (size= 1, demand= 138): C1
Job[2] (size= 1, demand= 188): C2
Job[3] (size= 2, demand= 287): C1 C2
Job[4] (size= 1, demand= 57): C1
Job[5] (size= 1, demand= 442): C1
Job[6] (size= 1, demand= 460): C1
Job[7] (size= 1, demand= 82): C1
Job[8] (size= 1, demand= 206): C1
Job[9] (size= 1, demand= 349): C2
Job[10] (size= 1, demand= 197): C1

EXAMPLE 3 Production Session 3

There are 12 Jobs in the file
Job[1](size= 5,demand= 130) : 2 3 5
Job[2](size= 4,demand= 235) : 4 1
Job[3](size= 6,demand= 433) : 1 4 2 5
Job[4](size= 5,demand= 308) : 5 4 3
Job[5](size= 6,demand= 124) : 1 2 4 5
Job[6](size= 7,demand= 395) : 2 3 5 1 4
Job[7](size= 6,demand= 487) : 4 3 4 2
Job[8](size= 6,demand= 213) : 2 5 2 4

Job[9](size= 4,demand= 78) : 3 1
Job[10](size= 5,demand= 224) : 4 5 3
Job[11](size= 6,demand= 179) : 3 1 3 4
Job[12](size= 6,demand= 102) : 1 4 2 1

Virtual_Cell[1] (size= 4, demand= 1232): 2 1 3 5
Virtual_Cell[2] (size= 3, demand= 1744): 1 4 3
Virtual_Cell[3] (size= 3, demand= 2314): 2 4 5

Job[1] (size= 2, demand= 130): C1 C3
Job[2] (size= 1, demand= 235): C2
Job[3] (size= 2, demand= 433): C2 C3
Job[4] (size= 2, demand= 308): C3 C2
Job[5] (size= 2, demand= 124): C1 C3
Job[6] (size= 2, demand= 395): C1 C3
Job[7] (size= 2, demand= 487): C2 C3
Job[8] (size= 1, demand= 213): C3
Job[9] (size= 1, demand= 78): C1
Job[10] (size= 2, demand= 224): C3 C1
Job[11] (size= 2, demand= 179): C1 C2
Job[12] (size= 2, demand= 102): C2 C1

EXAMPLE 3 Production Session 4

There are 9 Jobs in the file
Job[1](size= 4,demand= 251) : 3 2
Job[2](size= 4,demand= 439) : 4 3
Job[3](size= 4,demand= 223) : 5 3
Job[4](size= 4,demand= 38) : 4 3
Job[5](size= 6,demand= 213) : 1 5 4 5
Job[6](size= 4,demand= 268) : 2 4
Job[7](size= 5,demand= 54) : 2 3 2
Job[8](size= 4,demand= 336) : 1 3
Job[9](size= 4,demand= 134) : 2 3

Virtual_Cell[1] (size= 4, demand= 772): 1 3 5 4
Virtual_Cell[2] (size= 3, demand= 1184): 2 3 4

Job[1] (size= 1, demand= 251): C2
Job[2] (size= 1, demand= 439): C2
Job[3] (size= 1, demand= 223): C1
Job[4] (size= 1, demand= 38): C2
Job[5] (size= 1, demand= 213): C1
Job[6] (size= 1, demand= 268): C2
Job[7] (size= 1, demand= 54): C2
Job[8] (size= 1, demand= 336): C1
Job[9] (size= 1, demand= 134): C2

EXAMPLE 3 Production Session 5

There are 16 Jobs in the file
Job[1](size= 4,demand= 188) : 5 4
Job[2](size= 6,demand= 186) : 2 1 4 2
Job[3](size= 5,demand= 151) : 5 3 1
Job[4](size= 6,demand= 218) : 1 4 3 2
Job[5](size= 5,demand= 227) : 1 5 2
Job[6](size= 6,demand= 47) : 5 1 5 3
Job[7](size= 4,demand= 401) : 3 5
Job[8](size= 5,demand= 258) : 4 5 2
Job[9](size= 6,demand= 277) : 3 4 3 5
Job[10](size= 7,demand= 394) : 1 5 4 2 5
Job[11](size= 4,demand= 220) : 2 3
Job[12](size= 5,demand= 357) : 1 2 5
Job[13](size= 7,demand= 257) : 5 2 1 3 2
Job[14](size= 6,demand= 10) : 3 2 5 4
Job[15](size= 4,demand= 211) : 2 3
Job[16](size= 7,demand= 255) : 4 5 2 3 2

Virtual_Cell[1] (size= 4, demand= 3863): 1 2 5 3
Virtual_Cell[2] (size= 3, demand= 1323): 1 4 5
Virtual_Cell[3] (size= 2, demand= 277): 3 4
Virtual_Cell[4] (size= 2, demand= 186): 4 2

Job[1] (size= 1, demand= 188): C2
Job[2] (size= 2, demand= 186): C1 C4
Job[3] (size= 1, demand= 151): C1
Job[4] (size= 2, demand= 218): C2 C1
Job[5] (size= 1, demand= 227): C1
Job[6] (size= 1, demand= 47): C1
Job[7] (size= 1, demand= 401): C1
Job[8] (size= 2, demand= 258): C2 C1
Job[9] (size= 2, demand= 277): C3 C1
Job[10] (size= 3, demand= 394): C1 C2 C1
Job[11] (size= 1, demand= 220): C1
Job[12] (size= 1, demand= 357): C1
Job[13] (size= 1, demand= 257): C1
Job[14] (size= 2, demand= 10): C1 C2
Job[15] (size= 1, demand= 211): C1
Job[16] (size= 2, demand= 255): C2 C1

EXAMPLE 3 Production Session 6

There are 11 Jobs in the file
Job[1](size= 6,demand= 488) : 2 1 4 3
Job[2](size= 4,demand= 402) : 2 4
Job[3](size= 5,demand= 103) : 5 3 4
Job[4](size= 4,demand= 89) : 1 4
Job[5](size= 6,demand= 73) : 4 5 1 5
Job[6](size= 6,demand= 274) : 3 1 2 3
Job[7](size= 5,demand= 276) : 3 1 2
Job[8](size= 5,demand= 163) : 5 1 5
Job[9](size= 4,demand= 230) : 3 4
Job[10](size= 4,demand= 60) : 5 3
Job[11](size= 5,demand= 173) : 4 3 1

Virtual_Cell[1] (size= 4, demand= 2108): 2 3 4 1
Virtual_Cell[2] (size= 3, demand= 846): 1 5 3

Job[1] (size= 1, demand= 488): C1
Job[2] (size= 1, demand= 402): C1
Job[3] (size= 2, demand= 103): C2 C1
Job[4] (size= 1, demand= 89): C1
Job[5] (size= 2, demand= 73): C1 C2
Job[6] (size= 2, demand= 274): C2 C1
Job[7] (size= 1, demand= 276): C1
Job[8] (size= 1, demand= 163): C2
Job[9] (size= 1, demand= 230): C1
Job[10] (size= 1, demand= 60): C2
Job[11] (size= 2, demand= 173): C1 C2

EXAMPLE 3 Production Session 7

There are 15 Jobs in the file
Job[1](size= 7,demand= 356) : 1 4 5 4 3
Job[2](size= 4,demand= 161) : 5 3
Job[3](size= 5,demand= 181) : 1 3 5
Job[4](size= 5,demand= 438) : 5 1 4
Job[5](size= 6,demand= 327) : 4 3 5 2

Job[6](size= 5,demand= 130) : 3 5 3
Job[7](size= 7,demand= 324) : 5 2 4 2 3
Job[8](size= 5,demand= 375) : 5 2 3
Job[9](size= 5,demand= 325) : 3 4 3
Job[10](size= 6,demand= 475) : 3 2 1 5
Job[11](size= 5,demand= 210) : 2 5 1
Job[12](size= 8,demand= 300) : 3 1 5 1 2 3
Job[13](size= 8,demand= 188) : 3 4 2 3 4 3
Job[14](size= 6,demand= 361) : 3 2 5 3
Job[15](size= 5,demand= 334) : 1 2 4

Virtual_Cell[1] (size= 4, demand= 1655): 1 4 5 2
Virtual_Cell[2] (size= 3, demand= 3404): 1 2 3
Virtual_Cell[3] (size= 3, demand= 3345): 3 4 5

Job[1] (size= 2, demand= 356): C1 C2
Job[2] (size= 1, demand= 161): C3
Job[3] (size= 2, demand= 181): C2 C3
Job[4] (size= 1, demand= 438): C1
Job[5] (size= 2, demand= 327): C3 C1
Job[6] (size= 1, demand= 130): C3
Job[7] (size= 2, demand= 324): C1 C2
Job[8] (size= 2, demand= 375): C3 C2
Job[9] (size= 1, demand= 325): C3
Job[10] (size= 2, demand= 475): C2 C3
Job[11] (size= 2, demand= 210): C2 C1
Job[12] (size= 3, demand= 300): C2 C3 C2
Job[13] (size= 3, demand= 188): C3 C2 C3
Job[14] (size= 2, demand= 361): C2 C3
Job[15] (size= 2, demand= 334): C2 C3

EXAMPLE 3 Production Session 8

There are 12 Jobs in the file
Job[1](size= 6,demand= 324) : 3 4 3 2
Job[2](size= 4,demand= 476) : 2 1
Job[3](size= 7,demand= 143) : 4 5 3 5 3
Job[4](size= 6,demand= 331) : 5 4 3 5
Job[5](size= 7,demand= 323) : 5 2 1 2 1
Job[6](size= 4,demand= 431) : 1 2
Job[7](size= 6,demand= 347) : 1 4 2 3
Job[8](size= 5,demand= 212) : 2 4 5
Job[9](size= 5,demand= 202) : 3 1 5
Job[10](size= 6,demand= 87) : 2 4 3 4
Job[11](size= 4,demand= 493) : 5 4
Job[12](size= 7,demand= 57) : 1 3 4 5 2

Virtual_Cell[1] (size= 3, demand= 1933): 1 2 4
Virtual_Cell[2] (size= 2, demand= 671): 2 3
Virtual_Cell[3] (size= 4, demand= 2172): 3 5 4 1

Job[1] (size= 2, demand= 324): C3 C2
Job[2] (size= 1, demand= 476): C1
Job[3] (size= 1, demand= 143): C3
Job[4] (size= 1, demand= 331): C3
Job[5] (size= 2, demand= 323): C3 C1
Job[6] (size= 1, demand= 431): C1
Job[7] (size= 2, demand= 347): C1 C2
Job[8] (size= 2, demand= 212): C1 C3
Job[9] (size= 1, demand= 202): C3
Job[10] (size= 2, demand= 87): C1 C3
Job[11] (size= 1, demand= 493): C3
Job[12] (size= 2, demand= 57): C3 C1

EXAMPLE 3 Production Session 9

There are 10 Jobs in the file
Job[1](size= 5,demand= 467) : 2 3 1
Job[2](size= 5,demand= 29) : 3 1 4
Job[3](size= 5,demand= 338) : 3 1 5
Job[4](size= 6,demand= 313) : 4 2 3 5
Job[5](size= 4,demand= 327) : 3 5
Job[6](size= 6,demand= 128) : 1 2 3 2
Job[7](size= 4,demand= 136) : 1 5
Job[8](size= 4,demand= 130) : 5 2
Job[9](size= 6,demand= 58) : 1 4 3 5
Job[10](size= 6,demand= 165) : 3 1 2 4

Virtual_Cell[1] (size= 4, demand= 2062): 1 2 5 3
Virtual_Cell[2] (size= 3, demand= 565): 3 1 4

Job[1] (size= 1, demand= 467): C1
Job[2] (size= 1, demand= 29): C2
Job[3] (size= 1, demand= 338): C1
Job[4] (size= 2, demand= 313): C2 C1
Job[5] (size= 1, demand= 327): C1
Job[6] (size= 1, demand= 128): C1
Job[7] (size= 1, demand= 136): C1
Job[8] (size= 1, demand= 130): C1
Job[9] (size= 2, demand= 58): C2 C1
Job[10] (size= 2, demand= 165): C1 C2

EXAMPLE 3 Production Session 10

There are 14 Jobs in the file
Job[1](size= 6,demand= 200) : 5 2 4 1
Job[2](size= 7,demand= 301) : 2 3 5 1 5
Job[3](size= 4,demand= 383) : 3 1
Job[4](size= 6,demand= 347) : 4 1 5 3
Job[5](size= 7,demand= 345) : 1 5 1 2 1
Job[6](size= 7,demand= 191) : 3 4 5 1 4
Job[7](size= 4,demand= 270) : 3 4
Job[8](size= 6,demand= 256) : 3 5 1 4
Job[9](size= 7,demand= 130) : 5 1 2 4 5
Job[10](size= 4,demand= 419) : 5 1
Job[11](size= 6,demand= 479) : 5 1 2 5
Job[12](size= 6,demand= 121) : 2 1 3 2
Job[13](size= 4,demand= 164) : 4 3
Job[14](size= 7,demand= 299) : 1 4 5 4 3

Virtual_Cell[1] (size= 4, demand= 3088): 2 4 1 5
Virtual_Cell[2] (size= 3, demand= 805): 1 3 2
Virtual_Cell[3] (size= 2, demand= 924): 3 4
Virtual_Cell[4] (size= 2, demand= 603): 3 5

Job[1] (size= 1, demand= 200): C1
Job[2] (size= 2, demand= 301): C2 C1
Job[3] (size= 1, demand= 383): C2
Job[4] (size= 2, demand= 347): C1 C4
Job[5] (size= 1, demand= 345): C1
Job[6] (size= 2, demand= 191): C3 C1
Job[7] (size= 1, demand= 270): C3
Job[8] (size= 2, demand= 256): C4 C1
Job[9] (size= 1, demand= 130): C1
Job[10] (size= 1, demand= 419): C1
Job[11] (size= 1, demand= 479): C1
Job[12] (size= 2, demand= 121): C1 C2
Job[13] (size= 1, demand= 164): C3
Job[14] (size= 2, demand= 299): C1 C3

EXAMPLE 4 Production Session 1

There are 9 Jobs in the file
Jobs[1](size= 5,demand= 340) : 11 3 8
Jobs[2](size= 4,demand= 392) : 9 3
Jobs[3](size= 4,demand= 399) : 8 9
Jobs[4](size= 7,demand= 120) : 1 4 9 11 10
Jobs[5](size= 7,demand= 254) : 9 4 1 3 2
Jobs[6](size= 5,demand= 370) : 4 2 4
Jobs[7](size= 5,demand= 102) : 5 10 4
Jobs[8](size= 7,demand= 437) : 10 6 11 7 10
Jobs[9](size= 5,demand= 204) : 7 9 10

Virtual_Cell[1] (size= 4, demand= 766): 1 3 9 4
Virtual_Cell[2] (size= 2, demand= 102): 5 10
Virtual_Cell[3] (size= 6, demand= 1500): 7 10 8 9 6 11
Virtual_Cell[4] (size= 2, demand= 340): 11 3
Virtual_Cell[5] (size= 2, demand= 726): 2 4

Jobs[1] (size= 2, demand= 340): C4 C3
Jobs[2] (size= 1, demand= 392): C1
Jobs[3] (size= 1, demand= 399): C3
Jobs[4] (size= 2, demand= 120): C1 C3
Jobs[5] (size= 2, demand= 254): C1 C5
Jobs[6] (size= 1, demand= 370): C5
Jobs[7] (size= 2, demand= 102): C2 C5
Jobs[8] (size= 1, demand= 437): C3
Jobs[9] (size= 1, demand= 204): C3

EXAMPLE 4 Production Session 2

There are 9 Jobs in the file
Jobs[1](size= 4,demand= 198) : 8 3
Jobs[2](size= 7,demand= 144) : 7 5 3 6 8
Jobs[3](size= 7,demand= 240) : 10 7 4 8 7
Jobs[4](size= 4,demand= 66) : 11 3
Jobs[5](size= 5,demand= 295) : 10 8 9
Jobs[6](size= 7,demand= 400) : 1 8 5 1 7
Jobs[7](size= 5,demand= 109) : 8 10 7
Jobs[8](size= 6,demand= 172) : 1 7 10 6
Jobs[9](size= 7,demand= 113) : 8 6 8 1 11

Virtual_Cell[1] (size= 4, demand= 342): 8 3 7 5
Virtual_Cell[2] (size= 2, demand= 295): 8 9
Virtual_Cell[3] (size= 6, demand= 1364): 10 6 1 7 8 5
Virtual_Cell[4] (size= 4, demand= 349): 7 4 8 10
Virtual_Cell[5] (size= 2, demand= 179): 11 3

Jobs[1] (size= 1, demand= 198): C1
Jobs[2] (size= 2, demand= 144): C1 C3
Jobs[3] (size= 2, demand= 240): C3 C4
Jobs[4] (size= 1, demand= 66): C5
Jobs[5] (size= 2, demand= 295): C3 C2
Jobs[6] (size= 1, demand= 400): C3
Jobs[7] (size= 1, demand= 109): C4
Jobs[8] (size= 1, demand= 172): C3
Jobs[9] (size= 2, demand= 113): C3 C5

EXAMPLE 4 Production Session 3

There are 12 Jobs in the file
Jobs[1](size= 6,demand= 328) : 3 5 1 6
Jobs[2](size= 4,demand= 404) : 3 2
Jobs[3](size= 4,demand= 195) : 7 8
Jobs[4](size= 4,demand= 386) : 5 4
Jobs[5](size= 5,demand= 180) : 7 5 10
Jobs[6](size= 4,demand= 77) : 3 1
Jobs[7](size= 5,demand= 295) : 6 4 8
Jobs[8](size= 6,demand= 496) : 10 7 6 7
Jobs[9](size= 5,demand= 426) : 5 7 2
Jobs[10](size= 4,demand= 71) : 3 9
Jobs[11](size= 4,demand= 64) : 5 3
Jobs[12](size= 6,demand= 182) : 6 7 3 10

Virtual_Cell[1] (size= 7, demand= 2157): 3 1 5 10 6 7 2
Virtual_Cell[2] (size= 2, demand= 71): 3 9
Virtual_Cell[3] (size= 2, demand= 386): 5 4
Virtual_Cell[4] (size= 2, demand= 195): 7 8
Virtual_Cell[5] (size= 3, demand= 295): 6 4 8

Jobs[1] (size= 1, demand= 328): C1
Jobs[2] (size= 1, demand= 404): C1
Jobs[3] (size= 1, demand= 195): C4
Jobs[4] (size= 1, demand= 386): C3
Jobs[5] (size= 1, demand= 180): C1
Jobs[6] (size= 1, demand= 77): C1
Jobs[7] (size= 1, demand= 295): C5
Jobs[8] (size= 1, demand= 496): C1
Jobs[9] (size= 1, demand= 426): C1
Jobs[10] (size= 1, demand= 71): C2
Jobs[11] (size= 1, demand= 64): C1
Jobs[12] (size= 1, demand= 182): C1

EXAMPLE 4 Production Session 4

There are 14 Jobs in the file
Jobs[1](size= 4,demand= 43) : 1 4
Jobs[2](size= 4,demand= 243) : 1 10
Jobs[3](size= 5,demand= 303) : 7 5 8
Jobs[4](size= 5,demand= 397) : 6 10 6
Jobs[5](size= 4,demand= 456) : 3 2
Jobs[6](size= 4,demand= 339) : 1 2
Jobs[7](size= 6,demand= 262) : 7 11 3 7
Jobs[8](size= 6,demand= 362) : 3 5 9 6
Jobs[9](size= 6,demand= 455) : 4 5 4 5
Jobs[10](size= 5,demand= 310) : 2 10 3
Jobs[11](size= 6,demand= 460) : 10 6 1 10
Jobs[12](size= 4,demand= 349) : 10 9
Jobs[13](size= 6,demand= 131) : 9 8 10 4
Jobs[14](size= 5,demand= 109) : 1 3 11

Virtual_Cell[1] (size= 4, demand= 1635): 1 2 10 4
Virtual_Cell[2] (size= 2, demand= 766): 3 2
Virtual_Cell[3] (size= 4, demand= 674): 3 11 7 5
Virtual_Cell[4] (size= 5, demand= 1568): 3 5 9 6 10
Virtual_Cell[5] (size= 2, demand= 434): 9 8
Virtual_Cell[6] (size= 2, demand= 455): 4 5

Jobs[1] (size= 1, demand= 43): C1
Jobs[2] (size= 1, demand= 243): C1
Jobs[3] (size= 2, demand= 303): C3 C5
Jobs[4] (size= 1, demand= 397): C4
Jobs[5] (size= 1, demand= 456): C2
Jobs[6] (size= 1, demand= 339): C1
Jobs[7] (size= 1, demand= 262): C3
Jobs[8] (size= 1, demand= 362): C4
Jobs[9] (size= 1, demand= 455): C6
Jobs[10] (size= 2, demand= 310): C1 C2

Jobs[11] (size= 2, demand= 460): C4 C1
Jobs[12] (size= 1, demand= 349): C4
Jobs[13] (size= 2, demand= 131): C5 C1
Jobs[14] (size= 2, demand= 109): C1 C3

EXAMPLE 4 Production Session 5

There are 8 Jobs in the file
Jobs[1](size= 5,demand= 213) : 10 4 7
Jobs[2](size= 5,demand= 114) : 11 2 4
Jobs[3](size= 5,demand= 288) : 8 7 1
Jobs[4](size= 7,demand= 385) : 1 10 9 10 7
Jobs[5](size= 7,demand= 267) : 8 4 7 4 1
Jobs[6](size= 7,demand= 385) : 6 10 8 10 5
Jobs[7](size= 4,demand= 78) : 5 8
Jobs[8](size= 6,demand= 360) : 11 4 8 6

Virtual_Cell[1] (size= 4, demand= 1513): 8 4 7 10
Virtual_Cell[2] (size= 3, demand= 474): 11 2 4
Virtual_Cell[3] (size= 4, demand= 823): 6 10 8 5
Virtual_Cell[4] (size= 3, demand= 940): 1 10 9

Jobs[1] (size= 1, demand= 213): C1
Jobs[2] (size= 1, demand= 114): C2
Jobs[3] (size= 2, demand= 288): C1 C4
Jobs[4] (size= 2, demand= 385): C4 C1
Jobs[5] (size= 2, demand= 267): C1 C4
Jobs[6] (size= 1, demand= 385): C3
Jobs[7] (size= 1, demand= 78): C3
Jobs[8] (size= 3, demand= 360): C2 C1 C3

EXAMPLE 4 Production Session 6

There are 10 Jobs in the file
Jobs[1](size= 6,demand= 313) : 2 1 4 8
Jobs[2](size= 7,demand= 200) : 2 7 9 6 7
Jobs[3](size= 4,demand= 372) : 4 8
Jobs[4](size= 6,demand= 456) : 6 3 4 5
Jobs[5](size= 4,demand= 193) : 3 4
Jobs[6](size= 5,demand= 438) : 8 7 9
Jobs[7](size= 6,demand= 72) : 8 3 11 6
Jobs[8](size= 6,demand= 413) : 8 3 1 8
Jobs[9](size= 7,demand= 286) : 2 1 4 7 11
Jobs[10](size= 7,demand= 189) : 5 6 5 1 11

Virtual_Cell[1] (size= 4, demand= 638): 2 7 9 6
Virtual_Cell[2] (size= 5, demand= 1196): 3 11 6 4 5
Virtual_Cell[3] (size= 3, demand= 2307): 4 7 8
Virtual_Cell[4] (size= 2, demand= 599): 2 1
Virtual_Cell[5] (size= 2, demand= 413): 3 1
Virtual_Cell[6] (size= 3, demand= 189): 5 1 6

Jobs[1] (size= 2, demand= 313): C4 C3
Jobs[2] (size= 1, demand= 200): C1
Jobs[3] (size= 1, demand= 372): C3
Jobs[4] (size= 1, demand= 456): C2
Jobs[5] (size= 1, demand= 193): C2
Jobs[6] (size= 2, demand= 438): C3 C1
Jobs[7] (size= 2, demand= 72): C3 C2
Jobs[8] (size= 3, demand= 413): C3 C5 C3
Jobs[9] (size= 3, demand= 286): C4 C3 C2
Jobs[10] (size= 2, demand= 189): C6 C2

EXAMPLE 4 Production Session 7

There are 8 Jobs in the file
Jobs[1](size= 7,demand= 203) : 5 1 3 8 5
Jobs[2](size= 4,demand= 267) : 4 7
Jobs[3](size= 8,demand= 53) : 8 6 8 2 9 11
Jobs[4](size= 6,demand= 406) : 2 6 9 4
Jobs[5](size= 7,demand= 187) : 9 6 2 8 5
Jobs[6](size= 4,demand= 27) : 1 5
Jobs[7](size= 4,demand= 118) : 3 1
Jobs[8](size= 5,demand= 76) : 8 2 8

Virtual_Cell[1] (size= 2, demand= 53): 9 11
Virtual_Cell[2] (size= 4, demand= 535): 8 5 1 3
Virtual_Cell[3] (size= 2, demand= 406): 9 4
Virtual_Cell[4] (size= 2, demand= 267): 4 7
Virtual_Cell[5] (size= 4, demand= 722): 9 6 2 8

Jobs[1] (size= 1, demand= 203): C2
Jobs[2] (size= 1, demand= 267): C4
Jobs[3] (size= 2, demand= 53): C5 C1
Jobs[4] (size= 2, demand= 406): C5 C3
Jobs[5] (size= 2, demand= 187): C5 C2
Jobs[6] (size= 1, demand= 27): C2
Jobs[7] (size= 1, demand= 118): C2
Jobs[8] (size= 1, demand= 76): C5

EXAMPLE 4 Production Session 8

There are 18 Jobs in the file
Jobs[1](size= 4,demand= 50) : 11 7
Jobs[2](size= 8,demand= 150) : 11 8 7 3 10 9
Jobs[3](size= 4,demand= 82) : 8 4
Jobs[4](size= 6,demand= 313) : 9 2 6 8
Jobs[5](size= 4,demand= 152) : 5 11
Jobs[6](size= 5,demand= 341) : 7 8 6
Jobs[7](size= 7,demand= 399) : 3 4 9 4 6
Jobs[8](size= 6,demand= 36) : 9 7 10 3
Jobs[9](size= 6,demand= 177) : 5 10 6 2
Jobs[10](size= 6,demand= 264) : 6 8 9 5
Jobs[11](size= 6,demand= 179) : 9 8 3 7
Jobs[12](size= 4,demand= 54) : 4 8
Jobs[13](size= 7,demand= 287) : 2 9 8 9 5
Jobs[14](size= 6,demand= 156) : 8 5 2 7
Jobs[15](size= 4,demand= 131) : 7 1
Jobs[16](size= 7,demand= 45) : 7 5 10 11 9
Jobs[17](size= 4,demand= 242) : 10 2
Jobs[18](size= 4,demand= 232) : 7 5

Virtual_Cell[1] (size= 4, demand= 1952): 8 9 2 5
Virtual_Cell[2] (size= 3, demand= 1453): 8 6 4
Virtual_Cell[3] (size= 6, demand= 1057): 10 2 11 9 7 5
Virtual_Cell[4] (size= 2, demand= 177): 10 6
Virtual_Cell[5] (size= 2, demand= 131): 7 1
Virtual_Cell[6] (size= 4, demand= 1225): 8 7 3 4
Virtual_Cell[7] (size= 2, demand= 36): 10 3

Jobs[1] (size= 1, demand= 50): C3
Jobs[2] (size= 3, demand= 150): C3 C6 C3
Jobs[3] (size= 1, demand= 82): C2
Jobs[4] (size= 2, demand= 313): C1 C2
Jobs[5] (size= 1, demand= 152): C3
Jobs[6] (size= 2, demand= 341): C6 C2
Jobs[7] (size= 3, demand= 399): C6 C1 C2
Jobs[8] (size= 2, demand= 36): C3 C7
Jobs[9] (size= 3, demand= 177): C1 C4 C1

Jobs[10] (size= 2, demand= 264): C2 C1
Jobs[11] (size= 2, demand= 179): C1 C6
Jobs[12] (size= 1, demand= 54): C2
Jobs[13] (size= 1, demand= 287): C1
Jobs[14] (size= 2, demand= 156): C1 C6
Jobs[15] (size= 1, demand= 131): C5
Jobs[16] (size= 1, demand= 45): C3
Jobs[17] (size= 1, demand= 242): C3
Jobs[18] (size= 1, demand= 232): C3

EXAMPLE 4 Production Session 9

There are 10 Jobs in the file
Jobs[1](size= 4,demand= 318) : 11 8
Jobs[2](size= 6,demand= 414) : 3 11 5 2
Jobs[3](size= 6,demand= 177) : 5 7 5 2
Jobs[4](size= 6,demand= 176) : 7 4 2 4
Jobs[5](size= 4,demand= 284) : 2 8
Jobs[6](size= 6,demand= 24) : 7 2 8 1
Jobs[7](size= 5,demand= 105) : 11 1 7
Jobs[8](size= 6,demand= 193) : 8 7 5 9
Jobs[9](size= 6,demand= 317) : 7 1 2 10
Jobs[10](size= 5,demand= 164) : 6 9 8

Virtual_Cell[1] (size= 4, demand= 791): 5 2 7 4
Virtual_Cell[2] (size= 2, demand= 193): 5 9
Virtual_Cell[3] (size= 2, demand= 164): 6 9
Virtual_Cell[4] (size= 3, demand= 803): 7 1 8
Virtual_Cell[5] (size= 2, demand= 318): 11 8
Virtual_Cell[6] (size= 2, demand= 317): 2 10
Virtual_Cell[7] (size= 2, demand= 519): 3 11
Virtual_Cell[8] (size= 2, demand= 284): 2 8

Jobs[1] (size= 1, demand= 318): C5
Jobs[2] (size= 2, demand= 414): C7 C1
Jobs[3] (size= 1, demand= 177): C1
Jobs[4] (size= 1, demand= 176): C1
Jobs[5] (size= 1, demand= 284): C8
Jobs[6] (size= 2, demand= 24): C1 C4
Jobs[7] (size= 2, demand= 105): C7 C4
Jobs[8] (size= 2, demand= 193): C4 C2
Jobs[9] (size= 2, demand= 317): C4 C6
Jobs[10] (size= 2, demand= 164): C3 C4

EXAMPLE 4 Production Session 10

There are 9 Jobs in the file
Jobs[1](size= 5,demand= 324) : 4 10 7
Jobs[2](size= 5,demand= 45) : 9 5 1
Jobs[3](size= 4,demand= 72) : 5 4
Jobs[4](size= 6,demand= 205) : 8 4 1 10
Jobs[5](size= 4,demand= 324) : 3 11
Jobs[6](size= 6,demand= 253) : 5 7 2 5
Jobs[7](size= 7,demand= 126) : 5 2 5 2 8
Jobs[8](size= 6,demand= 185) : 7 2 9 6
Jobs[9](size= 4,demand= 388) : 7 2

Virtual_Cell[1] (size= 2, demand= 324): 3 11
Virtual_Cell[2] (size= 4, demand= 230): 5 1 9 6
Virtual_Cell[3] (size= 5, demand= 1348): 5 2 7 4 10
Virtual_Cell[4] (size= 4, demand= 331): 8 4 1 10

Jobs[1] (size= 1, demand= 324): C3
Jobs[2] (size= 1, demand= 45): C2
Jobs[3] (size= 1, demand= 72): C3
Jobs[4] (size= 1, demand= 205): C4
Jobs[5] (size= 1, demand= 324): C1
Jobs[6] (size= 1, demand= 253): C3
Jobs[7] (size= 2, demand= 126): C3 C4
Jobs[8] (size= 2, demand= 185): C3 C2
Jobs[9] (size= 1, demand= 388): C3

EXAMPLE 5 Production Session 1

There are 10 Jobs in the file
Job[1](size= 4,demand= 95) : 8 3
Job[2](size= 4,demand= 297) : 7 6
Job[3](size= 5,demand= 130) : 6 2 6
Job[4](size= 5,demand= 468) : 2 7 5
Job[5](size= 4,demand= 468) : 4 1
Job[6](size= 4,demand= 462) : 2 7
Job[7](size= 6,demand= 384) : 3 5 4 5
Job[8](size= 4,demand= 30) : 7 8
Job[9](size= 4,demand= 359) : 7 1
Job[10](size= 4,demand= 146) : 1 5

Virtual_Cell[1] (size= 3, demand= 1357): 2 6 7
Virtual_Cell[2] (size= 4, demand= 1825): 4 1 5 7
Virtual_Cell[3] (size= 2, demand= 30): 7 8
Virtual_Cell[4] (size= 2, demand= 479): 8 3

Job[1] (size= 1, demand= 95): C4
Job[2] (size= 1, demand= 297): C1
Job[3] (size= 1, demand= 130): C1
Job[4] (size= 2, demand= 468): C1 C2
Job[5] (size= 1, demand= 468): C2
Job[6] (size= 1, demand= 462): C1
Job[7] (size= 2, demand= 384): C4 C2
Job[8] (size= 1, demand= 30): C3
Job[9] (size= 1, demand= 359): C2
Job[10] (size= 1, demand= 146): C2

EXAMPLE 5 Production Session 2

There are 12 Jobs in the file
Job[1](size= 6,demand= 225) : 8 1 8 4
Job[2](size= 5,demand= 324) : 4 1 5
Job[3](size= 5,demand= 337) : 3 2 1
Job[4](size= 6,demand= 407) : 8 5 1 4
Job[5](size= 4,demand= 98) : 4 6
Job[6](size= 5,demand= 336) : 5 3 2
Job[7](size= 6,demand= 306) : 3 2 4 7
Job[8](size= 5,demand= 484) : 4 2 1
Job[9](size= 4,demand= 151) : 7 1
Job[10](size= 4,demand= 74) : 3 6
Job[11](size= 6,demand= 30) : 3 1 3 5
Job[12](size= 4,demand= 36) : 6 8

Virtual_Cell[1] (size= 2, demand= 366): 3 5
Virtual_Cell[2] (size= 2, demand= 74): 3 6
Virtual_Cell[3] (size= 5, demand= 1574): 4 1 8 6 5
Virtual_Cell[4] (size= 3, demand= 1493): 3 2 1
Virtual_Cell[5] (size= 2, demand= 306): 4 7
Virtual_Cell[6] (size= 2, demand= 151): 7 1

Job[1] (size= 1, demand= 225): C3
Job[2] (size= 1, demand= 324): C3
Job[3] (size= 1, demand= 337): C4

Job[4] (size= 1, demand= 407): C3
Job[5] (size= 1, demand= 98): C3
Job[6] (size= 2, demand= 336): C1 C4
Job[7] (size= 2, demand= 306): C4 C5
Job[8] (size= 2, demand= 484): C3 C4
Job[9] (size= 1, demand= 151): C6
Job[10] (size= 1, demand= 74): C2
Job[11] (size= 2, demand= 30): C4 C1
Job[12] (size= 1, demand= 36): C3

EXAMPLE 5 Production Session 3

There are 8 Jobs in the file
Job[1](size= 5,demand= 179) : 1 8 1
Job[2](size= 8,demand= 206) : 8 6 8 4 7 4
Job[3](size= 8,demand= 418) : 6 2 5 3 4 6
Job[4](size= 4,demand= 37) : 1 3
Job[5](size= 7,demand= 279) : 6 4 3 1 8
Job[6](size= 8,demand= 102) : 5 8 4 6 8 1
Job[7](size= 4,demand= 54) : 1 7
Job[8](size= 4,demand= 279) : 1 6

Virtual_Cell[1] (size= 2, demand= 316): 1 3
Virtual_Cell[2] (size= 2, demand= 54): 1 7
Virtual_Cell[3] (size= 4, demand= 520): 5 3 6 2
Virtual_Cell[4] (size= 4, demand= 1742): 8 4 6 1
Virtual_Cell[5] (size= 2, demand= 206): 7 4

Job[1] (size= 1, demand= 179): C4
Job[2] (size= 2, demand= 206): C4 C5
Job[3] (size= 2, demand= 418): C3 C4
Job[4] (size= 1, demand= 37): C1
Job[5] (size= 3, demand= 279): C4 C1 C4
Job[6] (size= 2, demand= 102): C3 C4
Job[7] (size= 1, demand= 54): C2
Job[8] (size= 1, demand= 279): C4

EXAMPLE 5 Production Session 4

There are 10 Jobs in the file
Job[1](size= 4,demand= 361) : 6 8
Job[2](size= 5,demand= 191) : 4 6 7
Job[3](size= 5,demand= 481) : 8 5 6
Job[4](size= 5,demand= 215) : 7 3 6
Job[5](size= 5,demand= 314) : 8 7 6
Job[6](size= 4,demand= 331) : 6 1
Job[7](size= 6,demand= 335) : 5 8 7 5
Job[8](size= 5,demand= 49) : 2 7 4
Job[9](size= 4,demand= 482) : 6 3
Job[10](size= 4,demand= 283) : 1 4

Virtual_Cell[1] (size= 3, demand= 1011): 7 3 6
Virtual_Cell[2] (size= 4, demand= 240): 7 4 6 2
Virtual_Cell[3] (size= 3, demand= 614): 1 4 6
Virtual_Cell[4] (size= 3, demand= 1491): 5 6 8
Virtual_Cell[5] (size= 2, demand= 335): 7 5

Job[1] (size= 1, demand= 361): C4
Job[2] (size= 1, demand= 191): C2
Job[3] (size= 1, demand= 481): C4
Job[4] (size= 1, demand= 215): C1
Job[5] (size= 2, demand= 314): C4 C1
Job[6] (size= 1, demand= 331): C3
Job[7] (size= 2, demand= 335): C4 C5
Job[8] (size= 1, demand= 49): C2
Job[9] (size= 1, demand= 482): C1
Job[10] (size= 1, demand= 283): C3

EXAMPLE 5 Production Session 5

There are 10 Jobs in the file
Job[1](size= 4,demand= 196) : 2 8
Job[2](size= 6,demand= 355) : 3 4 2 6
Job[3](size= 5,demand= 89) : 2 3 1
Job[4](size= 6,demand= 454) : 2 8 2 1
Job[5](size= 5,demand= 391) : 7 3 6
Job[6](size= 5,demand= 64) : 8 2 1
Job[7](size= 5,demand= 200) : 7 5 8
Job[8](size= 4,demand= 123) : 5 4
Job[9](size= 6,demand= 85) : 1 7 8 2
Job[10](size= 5,demand= 400) : 2 4 6

Virtual_Cell[1] (size= 3, demand= 799): 2 1 8
Virtual_Cell[2] (size= 5, demand= 1235): 2 4 3 6 1
Virtual_Cell[3] (size= 2, demand= 123): 5 4
Virtual_Cell[4] (size= 2, demand= 200): 5 8
Virtual_Cell[5] (size= 2, demand= 676): 1 7

Job[1] (size= 1, demand= 196): C1
Job[2] (size= 1, demand= 355): C2
Job[3] (size= 1, demand= 89): C2
Job[4] (size= 1, demand= 454): C1
Job[5] (size= 2, demand= 391): C5 C2
Job[6] (size= 1, demand= 64): C1
Job[7] (size= 2, demand= 200): C5 C4
Job[8] (size= 1, demand= 123): C3
Job[9] (size= 2, demand= 85): C5 C1
Job[10] (size= 1, demand= 400): C2

EXAMPLE 5 Production Session 6

There are 12 Jobs in the file
Job[1](size= 5,demand= 166) : 8 2 5
Job[2](size= 6,demand= 375) : 8 1 5 3
Job[3](size= 4,demand= 283) : 2 5
Job[4](size= 4,demand= 376) : 8 2
Job[5](size= 5,demand= 230) : 5 8 2
Job[6](size= 4,demand= 457) : 4 6
Job[7](size= 6,demand= 232) : 1 3 2 3
Job[8](size= 4,demand= 338) : 1 7
Job[9](size= 5,demand= 248) : 8 7 5
Job[10](size= 6,demand= 99) : 6 5 1 2
Job[11](size= 7,demand= 473) : 3 5 7 8 5
Job[12](size= 5,demand= 376) : 6 4 6

Virtual_Cell[1] (size= 5, demand= 1517): 1 2 3 5 7
Virtual_Cell[2] (size= 2, demand= 99): 6 5
Virtual_Cell[3] (size= 3, demand= 2151): 8 2 5
Virtual_Cell[4] (size= 2, demand= 248): 8 7
Virtual_Cell[5] (size= 2, demand= 833): 4 6

Job[1] (size= 1, demand= 166): C3
Job[2] (size= 2, demand= 375): C3 C1
Job[3] (size= 1, demand= 283): C3
Job[4] (size= 1, demand= 376): C3
Job[5] (size= 1, demand= 230): C3
Job[6] (size= 1, demand= 457): C5
Job[7] (size= 1, demand= 232): C1

Job[8] (size= 1, demand= 338): C1
Job[9] (size= 2, demand= 248): C4 C3
Job[10] (size= 2, demand= 99): C2 C1
Job[11] (size= 2, demand= 473): C1 C3
Job[12] (size= 1, demand= 376): C5

EXAMPLE 5 Production Session 7

There are 11 Jobs in the file
Job[1](size= 6,demand= 87) : 8 2 1 8
Job[2](size= 6,demand= 208) : 4 1 4 6
Job[3](size= 6,demand= 277) : 5 7 5 3
Job[4](size= 6,demand= 426) : 4 7 8 7
Job[5](size= 4,demand= 453) : 8 4
Job[6](size= 4,demand= 263) : 8 4
Job[7](size= 5,demand= 425) : 5 6 5
Job[8](size= 4,demand= 406) : 1 6
Job[9](size= 6,demand= 463) : 5 1 6 2
Job[10](size= 5,demand= 331) : 7 3 4
Job[11](size= 4,demand= 475) : 3 8

Virtual_Cell[1] (size= 5, demand= 1381): 5 1 6 2 8
Virtual_Cell[2] (size= 3, demand= 277): 5 3 7
Virtual_Cell[3] (size= 4, demand= 1948): 3 4 8 7
Virtual_Cell[4] (size= 3, demand= 208): 4 6 1

Job[1] (size= 1, demand= 87): C1
Job[2] (size= 1, demand= 208): C4
Job[3] (size= 1, demand= 277): C2
Job[4] (size= 1, demand= 426): C3
Job[5] (size= 1, demand= 453): C3
Job[6] (size= 1, demand= 263): C3
Job[7] (size= 1, demand= 425): C1
Job[8] (size= 1, demand= 406): C1
Job[9] (size= 1, demand= 463): C1
Job[10] (size= 1, demand= 331): C3
Job[11] (size= 1, demand= 475): C3

EXAMPLE 5 Production Session 8

There are 5 Jobs in the file
Job[1](size= 4,demand= 10) : 6 1
Job[2](size= 7,demand= 247) : 3 8 5 8 4
Job[3](size= 7,demand= 240) : 2 1 4 6 8
Job[4](size= 8,demand= 260) : 1 8 2 1 8 1
Job[5](size= 5,demand= 361) : 8 5 8

Virtual_Cell[1] (size= 2, demand= 10): 6 1
Virtual_Cell[2] (size= 3, demand= 487): 6 8 4
Virtual_Cell[3] (size= 3, demand= 608): 3 8 5
Virtual_Cell[4] (size= 3, demand= 500): 8 1 2

Job[1] (size= 1, demand= 10): C1
Job[2] (size= 2, demand= 247): C3 C2
Job[3] (size= 2, demand= 240): C4 C2
Job[4] (size= 1, demand= 260): C4
Job[5] (size= 1, demand= 361): C3

EXAMPLE 5 Production Session 9

There are 9 Jobs in the file
Job[1](size= 4,demand= 307) : 6 5
Job[2](size= 4,demand= 483) : 2 4
Job[3](size= 4,demand= 204) : 8 1
Job[4](size= 6,demand= 457) : 6 3 6 4
Job[5](size= 4,demand= 76) : 6 3
Job[6](size= 4,demand= 300) : 3 4
Job[7](size= 5,demand= 304) : 8 3 4
Job[8](size= 4,demand= 137) : 6 2
Job[9](size= 5,demand= 142) : 6 5 8

Virtual_Cell[1] (size= 4, demand= 1153): 6 4 2 3
Virtual_Cell[2] (size= 2, demand= 204): 8 1
Virtual_Cell[3] (size= 3, demand= 604): 8 3 4
Virtual_Cell[4] (size= 3, demand= 449): 6 5 8

Job[1] (size= 1, demand= 307): C4
Job[2] (size= 1, demand= 483): C1
Job[3] (size= 1, demand= 204): C2
Job[4] (size= 1, demand= 457): C1
Job[5] (size= 1, demand= 76): C1
Job[6] (size= 1, demand= 300): C3
Job[7] (size= 1, demand= 304): C3
Job[8] (size= 1, demand= 137): C1
Job[9] (size= 1, demand= 142): C4

EXAMPLE 5 Production Session 10

There are 14 Jobs in the file
Job[1](size= 6,demand= 76) : 4 1 5 1
Job[2](size= 6,demand= 87) : 7 6 8 3
Job[3](size= 6,demand= 93) : 2 4 7 4
Job[4](size= 4,demand= 79) : 8 3
Job[5](size= 6,demand= 136) : 2 7 1 4
Job[6](size= 4,demand= 473) : 8 4
Job[7](size= 6,demand= 395) : 6 8 6 5
Job[8](size= 5,demand= 409) : 5 4 3
Job[9](size= 4,demand= 249) : 1 8
Job[10](size= 4,demand= 69) : 1 4
Job[11](size= 4,demand= 244) : 8 4
Job[12](size= 5,demand= 237) : 1 8 1
Job[13](size= 6,demand= 468) : 4 3 5 8
Job[14](size= 6,demand= 479) : 8 2 8 2

Virtual_Cell[1] (size= 5, demand= 1570): 2 4 8 7 1
Virtual_Cell[2] (size= 4, demand= 1425): 8 1 5 6
Virtual_Cell[3] (size= 4, demand= 166): 8 3 7 6
Virtual_Cell[4] (size= 3, demand= 877): 5 4 3

Job[1] (size= 2, demand= 76): C1 C2
Job[2] (size= 1, demand= 87): C3
Job[3] (size= 1, demand= 93): C1
Job[4] (size= 1, demand= 79): C3
Job[5] (size= 1, demand= 136): C1
Job[6] (size= 1, demand= 473): C1
Job[7] (size= 1, demand= 395): C2
Job[8] (size= 1, demand= 409): C4
Job[9] (size= 1, demand= 249): C2
Job[10] (size= 1, demand= 69): C1
Job[11] (size= 1, demand= 244): C1
Job[12] (size= 1, demand= 237): C2
Job[13] (size= 2, demand= 468): C4 C2
Job[14] (size= 1, demand= 479): C1

APPENDIX G. VIRTUAL CELLS OBTAINED BY USING THE MODIFIED BOCTOR'S MODEL

To compare with virtual cells, Boctor's model is modified by allowing the concept of machine and cell sharing. The results obtained for each example is as follows:

EXAMPLE 1

	CELL 1		CELL 2		CELL 3		CELL 4		CELL 5		CELL 6		CELL 7	
	Machines	Vol.	Machines	Vol.	Machines	Vol.	Machines	Vol.	Machines	Vol.	Machines	Vol.	Machines	Vol.
EX11	10, 11, 12	190	1, 10	220	4, 7, 8, 9	855	2, 3, 5, 6,	435	10, 11, 12	0				
EX12	1, 11	120	4, 7, 8, 9	965	3, 6, 10, 12	435	1, 11	0	2, 11	200	5, 11	245		
EX13	1, 10, 11, 12	545	4, 7, 8, 9,	475	2, 3, 5, 6	0								
EX14	2, 10	0	2, 6, 7, 9	1465	3, 4, 5, 8,	1125	2, 10	0	1, 10, 11, 12	340				
EX15	3, 4, 5, 8	1025	7, 9	1450	2, 10	0	2, 10	0	1, 6, 11, 12	675				
EX16	1, 6, 7, 10	990	2, 4	0	3, 5, 8, 9	400	11, 12	390						
EX17	1, 2, 6, 10	450	11, 12	375	4, 7, 8, 9,	780	3, 5	0						
EX18	11, 12	170	1, 4, 6, 7, 8, 9,	1030	3, 5,	410	2, 10	100						
EX19	4, 7, 8, 9,	2905	3, 11	240	3, 11	0	3, 12	240	3, 11	0	1, 2, 6, 10	2095	3, 5	325
EX110	1, 2, 6	1000	1, 7, 10, 11, 12	1970	3, 5	505	2, 9	0	4, 8, 9	680				

EXAMPLE 2

	CELL 1		CELL 2		CELL 3		CELL 4		CELL 5		CELL 6		CELL 7	
	Machines	Vol.	Machines	Vol.	Machines	Vol.	Machines	Vol.	Machines	Vol.	Machines	Vol.	Machines	Vol.
EX21	3, 7,	434	1, 2, 4, 5, 6, 8	2284	3, 7	0	3, 7	0						
EX22	2, 5, 6, 7, 8	1954	1, 3	392	1, 4	355	1, 3	0	1, 3	0	1, 3	0		
EX23	4, 5, 7	474	2, 6, 8	912	5, 6	71	5, 6	0	1, 3	1080	5, 6	0		
EX24	7, 8	621	1, 2, 4, 5, 6,	3732	7, 8	0	3, 8	1539	7, 8	0				
EX25	2, 4, 5, 6, 7, 8	2161	1, 3	238	2, 4, 5, 6, 7, 8	0								
EX26	1, 3, 4, 5, 7, 8	771	1, 3, 4, 5, 7, 8	0	2, 6,	179								
EX27	1, 4,	272	5, 7,	620	5, 7,	0	4, 5,	217	3, 5, 6, 8	1635	5, 7	0	1, 7	343
EX28	1, 3, 4, 5, 7, 8,	1695	2, 6,	293										
EX29	2, 5, 6, 8	2343	1, 3, 4,	1094	4, 7,	1437	1, 4,	0	1, 6,	0				
EX210	2, 5	0	1, 2, 5, 6, 7, 8	2804	1, 3, 4, 5, 6, 8	2236	1, 3, 4, 5, 6, 8	0	3, 5	629	1, 3, 4, 5, 6, 8	0		

	CELL 8	
	Machines	Vol.
EX21		
EX22		
EX23		
EX24		
EX25		
EX26		
EX27	1, 2, 4, 8	1496
EX28		
EX29		
EX210		

EXAMPLE 3

	CELL 1		CELL 2		CELL 3		CELL 4	
	Machines	Vol.	Machines	Vol.	Machines	Vol.	Machines	Vol.
EX31	1, 2, 3,	1913	4, 5	2438	1, 4	432		
EX32	1, 4	770	2, 3, 4, 5	2324				
EX33	1, 2, 4, 5	2830	1, 3	1406	1, 5	395		
EX34	1, 5	687	1, 2, 3, 4,	1956				
EX35	1, 4	2013	1, 4	0	1, 2, 3, 5	3657	1, 4	0
EX36	1, 2, 3, 4	2168	1, 5	399				
EX37	2, 3, 4, 5	4485	1, 4	2294	1, 4	0		
EX38	4, 5	1761	3, 4	1491	1, 2	2459		
EX39	4, 5	565	1, 2, 3, 5	2091				
EX310	2, 3	1455	2, 3	0	2, 3	0	1, 3, 4, 5	3905

EXAMPLE 4

	CELL 1		CELL 2		CELL 3		CELL 4		CELL 5		CELL 6	
	Machines	Vol.	Machines	Vol.	Machines	Vol.	Machines	Vol.	Machines	Vol.	Machines	Vol.
EX41	5, 6, 7, 10, 11	1203	1, 2, 3, 4, 8, 9	2181	1, 5	0	1, 5	0				
EX42	2, 9	295	1, 5, 6, 7, 8, 10	1671	2, 3	408	2, 11	179	2, 4	240		
EX43	4, 8	876	1, 11	405	2, 3, 5, 6, 7, 10	3104	9, 11	71	9, 11	0		
EX44	4, 5, 8	1294	1, 6, 9, 10	2743	7, 11	565	8, 11	240	2, 3,	1576		
EX45	2, 3,	114	3, 9	385	1, 4, 7, 10	2012	5, 6, 8, 11	1492				
EX46	2, 10	799	10, 11	547	7, 9, 10	924	6, 10	917	10, 11	0	1, 3, 4, 5, 8	2732
EX47	7, 10	267	10, 11	53	2, 4, 6, 9	989	10, 11	0	1, 3, 5, 6	664		
EX48	1, 7	526	1, 11	150	1, 10	650	1, 11	0	5, 7, 11	1728	2, 3, 4, 6, 8, 9,	2725
EX49	6, 9	357	6, 10	317	6, 10	0	1, 2, 4	1497	6, 10	0	6, 10	0
EX410	2, 5, 6, 7, 9	1393	3, 11	324	1, 4, 8, 10	772	6, 9	185				

	CELL 7		CELL 8	
	Machines	Vol.	Machines	Vol.
EX41				
EX42				
EX43				
EX44				
EX45				
EX46				
EX47				
EX48	1, 11	0		
EX49	6, 7, 10	992	3, 5, 8, 11	1679
EX410				

EXAMPLE 5

	CELL 1		CELL 2		CELL 3		CELL 4		CELL 5	
	Machines	Vol.	Machines	Vol.	Machines	Vol.	Machines	Vol.	Machines	Vol.
EX51	3, 8	125	3, 8	0	2, 6, 7	1746	1, 3, 4, 5	1825		
EX52	1, 2, 3, 4, 5,	2772	6, 7	629	6, 7	0	6, 7	0	6, 8	668
EX53	1, 4, 6, 8	1554	2, 7	678	5, 7	520	3, 7	734	2, 7	0
EX54	2, 4	240	2, 4	0	1, 3, 6, 7,	2759	5, 8	1491	1, 4	283
EX55	1, 2, 4, 6, 8	2234	5, 7	285	4, 5	123	3, 7	835	5, 7	0
EX56	1, 4	331	2, 3, 5, 7, 8	2151	1, 7	713	4, 6	932	1, 4	0
EX57	2, 3	0	3, 4, 7, 8	2520	1, 2, 5, 6	1866	2, 3	0		
EX58	4, 7	487	4, 7	250	1, 2, 5, 8	1118	3, 7	247		
EX59	2, 3, 4, 6	2206	1, 5, 8	957	1, 7	0	1, 7	0		
EX510	5, 6	1348	6, 7	316	6, 7	0	1, 2, 3, 4, 8	3494		

REFERENCES

1) Sule, D. R., Manufacturing Facilities: Location, Planning, and Design, 2nd ed., PWS, Boston, MA., 1994.

2) Browne, J., Harhen, J., and Shivnan, J., Production Management Systems, Addison-Wesley, NY., 1988.

3) Tompkins, J. A., *et al.*, Facilities Planning, 2nd ed., John Wiley and Sons, NY, 1996.

4) Drolet, J., Abdulnour, G., and Rheault, M., "The Cellular Manufacturing Evolution," Proceeding of 19th Conference on Computers and Industrial Engineering, Vol. 31, No. 1/2, 1996, pp. 139-142.

5) Flynn, B. B., and Jacobs, F. R., "A simulation comparison of group technology with traditional job shop manufacturing," International Journal of Production Research, Vol. 24, No. 5, 1986, pp. 1171-1192.

6) Flynn, B. B., and Jacobs, F. R., "An experimental comparison of cellular layout with process layout," Decision Sciences, Vol. 18, No. 4, 1987, pp. 562-581.

7) Morris, J. S., and Tersine, R. J., "A simulation analysis of factors influencing the attractiveness of group technology cellular layouts," Management Science, Vol. 36, No. 12, 1990, pp. 1567-1578.

8) Morris, J. S., and Tersine, R. J., "A simulation comparison of process and cellular layouts in a dual resource constrained environment," Computer and Industrial Engineering, Vol. 26, No. 4, 1994, pp. 733-741.

9) Jensen, J. B., Malhotra, M. K., and Philipoom, P. R., "Machine dedication and process flexibility in a group technology," Journal of Operations Management, Vol. 14, 1996, pp. 19-39.

10) Shafer, S. M., and Charnes, J. M., "Cellular versus functional layouts under a variety of shop operating conditions," Decision Sciences, Vol. 24, No. 3, 1993, pp. 665-681.

11) Shafer, S. M., and Charnes, J. M., "A simulation analysis of factors influencing loading practices in cellular manufacturing," International Journal of Production Research, Vol. 33, No. 1, 1995, pp. 279-290.

12) Wemmerlov, U., and Vakharia, A. J., "Job and family scheduling of a flow-line manufacturing cell: A simulation study," IIE Transactions, Vol. 23, No. 4, 1991, pp. 383-393.

13) Burgess, A. G., Morgan I., and Vollmann T. E., "Cellular manufacturing: its impact on the total factory," International Journal of Production Research, Vol. 31, No. 9, 1993, pp. 2059-2077.

14) Garza, O., and Smunt, T. L., "Countering the negative impact of intercell flow in cellular manufacturing," Journal of Operations Management, Vol. 10, No. 1, 1991, 91-118.

15) Suresh, N. C., "Partitioning work centers for group technology: Insights from an analytical model," Decision Sciences, Vol. 22, No. 4, 1991, PP. 772-791.

16) Suresh, N. C., "Partitioning work centers for group technology: Analytical extension and shop-level simulation investigation," Decision Sciences, Vol. 23, No. 2, 1992, pp. 267-290.

17) Suresh, N. C., and Meredith, J. R., "Coping with the loss of pooling synergy in cellular manufacturing systems," Management Science, Vol. 40, No. 4, 1994, pp. 466-483.

18) Suresh, N. C., "Evaluation of functional and cellular layouts through simulation and analytical models," (Editors: Suresh, N. C., and Kay, J. M.), Kluwer Academic Publishers, MA, 1998, pp. 273-288.

19) Agarwal, A., Huq, F., and Sarkis, J., "Performance of manufacturing cells for group technology: a parametric analysis," Planning, Design, and Analysis of Cellular Manufacturing Systems (Editors: Kamrani *et al.*), Elsevier, NY, 1995, pp. 167-180.

20) Kannan, V. R., and Ghosh, Soumen, "Cellular Manufacturing Using Virtual Cells," International journal of Operations and Production Management, Vol. 16, No. 5, 1996, pp. 99-112.

21) McLean, C. R., Bloom, H. M., and Hopp, T. H., "The Virtual Manufacturing Cell," Proceedings of 4th IFAC/IFIP Conference on Information Control Problems in Manufacturing Technology, Maryland, pp. 207-215., 1982

22) Drolet, J. R., Montreuil, B., and Moodie, C. L., "Virtual cellular manufacturing layout planning," Proceedings of 19th International Industrial Engineering Conference, 1990, pp. 236-241.

23) Goetz, W. G. Jr., and Egbelu P. J., "Guide path design and location of load pick-up/drop-off points for an automated guided vehicle system," International Journal of Production Research, Vol. 28, No.5, 1990, pp. 927-941.

24) Kaspi, M., and Tanchoco, J. M. A., "Optimal flow design of unidirectional AGV systems," International Journal of Production Research, Vol. 28, No. 6, 1990, pp. 1023-1030.

25) Gaskins, R. J., and Tanchoco, J. M. A., "Flow path design for automated guided vehicle systems," International Journal of Production Research, Vol. 25, No. 5, 1987, pp. 667-676.

26) Venkataramanan, M. A., and Wilson K. A., "A Branch-and-Bound Algorithm for Flow-Path Design of Automated Guided Vehicle Systems," Naval Research Logistics, Vol. 38, 1991, pp. 431-445.

27) Seo, Y., and Egbelu, P. J., "Flexible guidepath design for automated guided vehicle systems," International Journal of Production Research, Vol. 33, No. 4, 1995, pp. 1135-1156.

28) Kouvelis, P., Gutierrez, G. J., and Chiang, W. C., "Heuristic unidirectional flowpath design approach for automated guided vehicle systems," International Journal of Production Research, Vol. 30, No. 6, 1992, pp. 1327-1351.

29) Burbidge, J. "The introduction of group technology," Heinemann, London, 1975.

30) Wemmerlöv, U., and Hyer, N. "Procedures for the part family/ machine group identification problem in cellular manufacturing," Journal of Operations Management, Vol. 6, No. 2, 1986, pp. 125-147.

31) Hyer, N. "The potential of group technology for US manufacturing," Journal of operations Management, Vol. 4, No. 3, 1984, pp. 183-202.

32) Selim, H. M., Askin, R. G., and Vakharia, A. J., "Cell formation in group technology: Review, Evaluation and Directions for future research," Computers and Industrial Engineering, Vol. 34, No. 1, 1998, pp. 3-20.

33) Burbidge, J. L., "Production flow analysis," Production Engineer, December 1963, pp. 742-752.

34) Burbidge, J. L., "Production flow analysis," Production Engineer, April/May 1971, pp. 139-152.

35) Burbidge, J. L., "A manual method of production flow analysis," Production Engineer, October 1977, pp. 34-38.

36) Karvonen, S., Holmström, J., and Eloranta, E., "Benefits from PFA in two make-to-order manufacturing firms in Finland," in Group Technology and Cellular Manufacturing (Editors: Suresh, N. C., and Kay, J. M.), Kluwer Academic Publishers, MA, 1998, pp. 458-474.

37) El-Essawy, I. G., and Torrance, J., "Component flow analysis – an effective approach to production systems' design," Production Engineer, May 1972, pp. 165-170.

38) Shafer, S. M., "Part-machine-labor grouping: the problem and solution methods," in Group Technology and Cellular Manufacturing (Editors: Suresh, N. C., and Kay, J. M.), Kluwer Academic Publishers, MA, 1998, pp. 131-152.

39) Gettleman, K. M., "Organize production for parts – not processes," Modern Machine Shop, November, 1971, pp. 50-60.

40) Allison, J. W., and Vapor, R., "GT approach proves out," American Machinist, February, 1979, pp. 86-89.

41) Shafer, S. M., Meredith, J. R., and Marsh, R. F., "A comparison of cellular manufacturing research assumptions with practice," Working Paper, Auburn University, USA, 1997.

42) Irani, S. A., A flow decomposition approach for integrated layout design of virtual group technology flowlines, Ph. D. dissertation, The Pennsylvania State University, 1990.

43) Chu, C. H., "Recent advances in mathematical programming for cell formation," in Planning, Design, and Analysis of Cellular Manufacturing Systems (Editors: Kamrani *et al.*), Elsevier, NY, 1995, pp. 3-46.

44) Boctor, F. F., "A linear formulation of the machine-part cell formation problem," International Journal of Production Research, Vol. 29, 1991, pp. 343-356.

45) Dahel, N. E., and Smith, S. B., "Designing flexibility into cellular manufacturing systems," International Journal of Production Research, Vol. 31, 1993, pp. 933-945.

46) Shafer, S. M., and Meredith, J. R., "A comparison of selected manufacturing cell formation techniques," International Journal of Production Research, Vol. 28, 1990, pp. 661-673.

47) Shafer, S., and Rogers, D. F., "A goal programming approach to the cell formation problem," Journal of Operations Management, Vol. 10, No. 1, 1991, pp. 28-43.

48) Wei, J. C., and Gaither, N., "An optimal model for cell formation decisions," Decision Sciences, Vol. 21, No. 2, 1990, pp.416-433.

49) Sankaran, S., and Kasilingam, R. G., "On cell size and machine requirements planning in group technology systems," European Journal of Operational Research, Vol. 69, No. 3, 1993, pp. 373-383.

50) Chen, M. Y., "A mathimatical programming model for system reconfigruation in a dynamic cellular manufacturing environment," Annals of Operations Research, Vol. 77, 1998, pp. 109-128.

51) Askin, R. G., and Chiu, K. S., "A graph partitioning procedure for machine assignment and cell formation in group technology," International Journal of Production Research, Vol. 28, No. 8, 1990, pp. 1555-1572.

52) Gunasingh, K. R., and Lashkari, R. S., "Machine grouping problem in cellular manufacturing systems – an integer programming approach," International Journal of Production Research, Vol. 27, No. 9, 1989, pp. 1465-1473.

53) Choobineh, F., "A framework for the design of cellular manufacturing systems," International Journal of Production Research, Vol. 26, 1988, pp. 1161-1172.

54) Kusiak, A., "The generalized group technology problem," International Journal of Production Research, Vol. 25, 1987, pp. 561-569.

55) Ferreira, J. F., Ribeiro, C., and Pradin, B., "A methodology for cellular manufacturing design," International Journal of Production Research, Vol. 13, 1975, pp. 56-68.

56) Ballakur, A., and Steudel, H. J., "A within-cell utilization based heuristic for designing cellular manufacturing systems," International Journal of Production Research, Vol. 25, 1987, pp. 639-665.

57) Co, H. C., and Araar, A., "Configuring cellular manufacturing systems," International Journal of Production Research, Vol. 26, No. 9, 1988, pp. 1511-1522.

58) Shtub, A., "Modeling group technology cell formation as a generalized assignment problem," International Journal of Production Research, Vol. 27, No. 5, 1989, pp. 775-782.

59) Zhou, M., and Askin, R. G., "Formation of general GT cells: an operation-based approach," Computers and Industrial Engineering, Vol. 34, No. 1, 1998, pp. 147-157.

60) Rardin, R. L., "Discrete optimization," in Handbook of Industrial Engineering, (Editor: Salvendy, G.), Wiley Interscience, Chichester, 1992.

61) Moodie, C. L., Drolet, J., Ho, Y. C., and Warren, G. M. H., "Cell design strategies for efficient material handling," in Material flow systems in manufacturing (Editor: Tanchoco, J. M. A.), Chapman & Hall, London, 1994.

62) Rajagopalan, R., and Batra, J., "Design of cellular production systems – a graph theoretic approach," International Journal of Production Research, Vol. 13, 1975, pp. 56-68.

63) Sokal, R. R., and Sneath, P. H. A., Principles of Numerical Taxonomy, Freeman, 1968.

64) McAuley, J., "Machine grouping for efficient production," Production Engineer, Vol. 51, 1972, pp. 53-57.

65) De Witte, J., "The use of similarity coefficients in production flow analysis," International Journal of Production Research, Vol. 18, No. 4, 1980, pp. 502-514.

66) Kusiak, A., and Chow, W. S., "Decomposition of manufacturing systems," IEEE Journal of Robotics and Automation, Vol. 4, No. 5, 1988, pp. 457-471.

67) Singh, N., and Rajamani, D., Cellular manufacturing systems – design, planning, and control, Chapman & Hall, London, 1996.

68) Lee, J. L., Vogt, W. G., and Mickle, M. H., "Calculation of shortest paths by optimal decomposition," IEEE Transactions on Systems, Man, and Cybernetics, Vol. 12, 1982, pp. 410-415.

69) Vannelli, A., and Kumar, K. R., "A method for finding minimal bottleneck cells for grouping part-machine families," International Journal of Production Research, Vol. 24, No. 2, 1986, pp. 387-400.

70) Joines, J. A., King, R. E., and Culbreth, C. T., "A comprehensive review of production-oriented manufacturing cell formation techniques," Working paper, North Carolina State University, USA, 1996.

71) Vohra, T., Chen, D. S., Chang, J. C., and Chen, H. C., "A network approach to cell formation in cellular manufacturing," International Journal of Production Research, Vol. 28, No. 11, 1990, pp. 2075-2084.

72) Wu, N., and Salvendy, G., "A modified network approach for the design of cellular manufacturing systems," International Journal of Production Research, Vol. 31, No. 6, pp. 1409-1421.

73) Lee, H., and Garcia-Diaz, A., "A network flow approach to solve clustering problems in group technology," International Journal of Production Research, Vol. 31, 1993, pp. 603-612.

74) Lee, H., and Garcia-Diaz, A., "Network flow partitioning procedures for the analysis of cellular manufacturing systems," Working paper, Texas A&M University, USA, 1993.

75) Garcia-Diaz, A., and Lee, Hongchul, "An industrial application of network-flow models in cellular manufacturing planning," in Planning, Design, and Analysis of Cellular Manufacturing Systems (Editors: Kamrani *et al.*), Elsevier, NY, 1995, pp. 47-61.

76) Ng, M. S., "Worst-case analysis of an algorithm for cellular manufacturing," European Journal of Operational Research, Vol. 69, No. 3, 1993, pp. 384-398.

77) Askin, R., and Chiu, K., "A graph partitioning procedure for machine assignment and cell formation," International Journal of Production Research, Vol. 28, No. 8, 1990, pp. 1555-1572.

78) Singh, N., "Design of cellular manufacturing systems: an invited review," European Journal of Operational Research, Vol. 69, 1993, pp. 284-291.

79) Chu, C. H., and Tsai, M., "A comparison of three array-based clustering techniques for manufacturing cellular formation," International Journal of Production Research, Vol. 28, 1990, pp. 1417-1433.

80) McCormick, W. T., Schweitzer, P. J., and White, T. W., "Problem decomposition and data reorganization by a cluster technique," Operations Research, Vol. 20, No. 5, 1972, pp. 993-1009.

81) King, J. R., "Machine-component grouping formation in group technology," International Journal of Management Science, Vol. 8, No. 2, 1980, pp. 193-199.

82) King, J. R., "Machine-component grouping in production flow analysis: An approach using a rank order clustering algorithm," International Journal of Production Research, Vol. 18, No. 2, 1980, pp. 213-232.

83) King, J. R., and Nakornchai, V., "Machine component group formation in group technology – review and extension," International Journal of Production Research, Vol. 20, pp. 117-133.

84) Chandrasekharan, M. P., and Rajagopalan, R., "GROUPABILITY: an analysis of the properties of binary data for group technology," International Journal of Production Research, Vol. 27, 1989, pp. 1035-1052.

85) Chan, H. M., and Milner, D. A., "Direct clustering algorithm for group formation in cellular manufacturing," Journal of Manufacturing Systems, Vol. 1, 1982, pp. 65-74.

86) Khator, S. M., and Irani, S. A., "Cell formation in group technology: a new approach," Computers and Industrial Engineering, Vol. 12, 1987, pp. 131-142.

87) Kusiak, A., and Chow, W. S., "Efficient solving of the group technology problem," Journal of Manufacturing Systems, Vol. 6, 1987, pp. 117-124.

88) Logendran, R., "A workload based model for minimizing total intercell and intracell moves in cellular manufacturing," International Journal of Production Research, Vol. 28, 1990, pp. 913-925.

89) Askin, R. G., Creswell, J. B., Goldberg, J. B., and Vakharia, A. J., "A hamiltonian path approach to reordering the part-machine matrix for cellular manufacturing," International Journal of Production Research, Vol. 29, 1991, pp. 1081-1100.

90) Stanfel, L. E., "Machine clustering for economic production," Engineering Costs and Production Economics, Vol. 9, 1985, pp. 73-81.

91) Mosier, C. T., and Taube, L., "Weighted similarity measure heuristics for the group technology machine clustering problem," International Journal of Management Science, Vol. 13, No. 6, 1985, pp. 577-583.

92) Seifoddini, H., and Djassemi, M., "Merits of the production volume based similarity coefficient in machine cell formation," Journal of Manufacturing Systems, Vol. 14, No. 1, 1995, pp. 35-44.

93) Gupta, T., and Seifoddini, H., "Production data based similarity coefficient for machine-component grouping decisions in the design of a cellular manufacturing system," International Journal of Production Research, Vol. 28, No. 7, 1990, pp. 1247-1269.

94) Choobineh, F., "A framework for the design of cellular manufacturing systems," International Journal of Production Research, Vol. 26, No. 7, 1988, pp. 1161-1172.

95) Tam, K. Y., "An operation sequence based similarity coefficient for part families formations," Journal of Manufacturing Systems, Vol. 9, No. 1, 1990, pp. 55-58.

96) Moussa, S. E., and Kamel, M. S., "A direct method for cell formation and part-machine assignment based on operation sequences and processing time similarity," Engineering Design and Automation, Vol. 2, No. 2, 1996, pp. 141-155.

97) Vakharia, A. J., and Wemmerlov, U., "Designing a cellular manufacturing system: a material flow approach based on operations sequence," IIE Transactions, Vol. 22, No. 1, 1990, pp. 84-97.

98) Selvam, R. P., and Balasubramanian, K. N., "Algorithmic grouping of operation sequences," Engineering Costs and Production Economics, Vol. 9, 1985, pp. 125-134.

99) Gunasingh, K. R., and Laskari, R. S., "The cell formation problem in cellular manufacturing systems – a sequential modeling approach," Computers and Industrial Engineering, Vol. 16, No. 4, 1989, pp. 469-476.

100) Gupta, T., "Design of manufacturing cells for flexible environment considering alternative routing," International Journal of Production Research, Vol. 31, No. 6, 1993, pp. 1259-1273.

101) Jeon, G., Leep, H. R., and Parsaei, H. R., "A cellular manufacturing system based on new similarity coefficient which considers alternative routes during machine failure," Computers and Industrial Engineering, Vol. 34, No. 1, 1998, pp. 21-36.

102) Kamrani, A. K., and Parsaei, H. R., "A group technology based methodology for machine cell formation in a computer integrated manufacturing environment," Computers and Industrial Engineering, Vol. 24, No. 3, 1993, pp. 431-447.

103) Seifoddini, H., and Hsu, C. P., "Comparative study of similarity coefficients and clustering algorithms in cellular manufacturing," Journal of Manufacturing Systems, Vol. 13, No. 2, 1994, pp. 119-127.

104) Nair, G. J., and Narendran, T. T., "Case: a clustering algorithm for cell formation with sequence data," International Journal of Production Research, Vol. 36, No. 1, 1998, pp. 157-179.

105) Wei, J. C., and Kern, G. M., "Commonality analysis: a linear cell clustering algorithm for group technology," International Journal of Production Research, Vol. 27, No. 12, 1989, pp. 2053-2062.

106) Kusiak, A., and Cho, M., "Similarity coefficient algorithms for solving the group technology problem," International Journal of Production Research, Vol. 30, No. 11, 1992, pp. 2633-2646.

107) Dutta, S. P., *et al.*, "A heuristic procedure for determining manufacturing families from design-based grouping for flexible manufacturing systems," Computers and Industrial Engineering, Vol. 10, 1986, pp. 193-201.

108) Cheng, C. H., *et al.*, "A TSP-based heuristic for forming machine groups and part families," International Journal of Production Research, Vol. 36, No. 5, 1998, pp. 1325-1337.

109) Seifoddini, H., "Single linkage versus average linkage clustering in machine cell formation applications," Computers and Industrial Engineering, Vol. 16, 1989, pp. 419-426.

110) Mosier, C. T., "An experiment investigating the application of clustering procedures and similarity coefficients to the GT machine cell formation problem," International Journal of Production Research, Vol. 27, No. 10, 1989, pp. 1811-1835.

111) Gupta, T., "Clustering algorithms for the design of a cellular manufacturing system – an analysis of their performance," Computers and Industrial Engineering, Vol. 20, No. 4, 1991, pp. 343-353.

112) Chandrasekharan, M. P., and Rajagopalan, R., "An ideal seed non-hierarchical clustering algorithm for cellular manufacturing," International Journal of Production Research, Vol. 24, No. 2, 1986, pp. 451-464.

113) MacQueen, J. B., "Some methods for classification and analysis of multivariate observations," in 5th Symposium on Mathematical Statistics and Probability, University of California, USA, 1967, p. 281.

114) Chandrasekharan, M. P., and Rajagopalan, R., "Zodiac – an algorithm for concurrent formation of part families and machine cells," International Journal of Production Research, Vol. 25, No. 6, 1987, pp. 835-850.

115) Srinivasan, G., and Narendran, T. T., "GRAFICS – a nonhierarchical clustering algorithm for group technology," International Journal of Production Research, Vol. 29, No. 3, 1991, pp. 463-478.

116) Kumar, K. R., and Chandrasekharan, M. P., "Grouping efficacy: a quantitative criterion for goodness of block diagonal forms of binary matrices in group technology," International Journal of Production Research, Vol. 28, No. 2, 1990, pp. 233-243.

117) Miltenburg, J., and Zhang, W., "A comparative evaluation of nine well-known algorithms for solving the cell formation in group technology," Journal of Operations Management, Vol. 10, No. 1, 1991, pp. 44-72.

118) Wu, H, Venugopal, M., and Barash, M., "Design of a cellular manufacturing system: A syntactic pattern recognition approach," Journal of Manufacturing Systems, Vol. 5, No. 2, 1986, pp. 81-88.

119) Tam, K. Y., "Linguistic modeling of flexible manufacturing systems," Journal of Manufacturing Systems, Vol. 8, No. 2, 1989, pp. 127-137.

120) Kusiak, A., "EXGT-S: A knowledge based system for group technology," International Journal of Production Research, Vol. 26, 1988, pp. 887-904.

121) ElMaraghy, H. A., and Gu, P., "Feature based expert parts assignment in cellular manufacturing," International Journal of Advanced Manufacturing Technology, Vol. 8, 1988, pp. 139-152.

122) Singh, N., and Qi, D. Z., "A syntactic pattern recognition based approach to the design of cellular manufacturing systems with multi-dimensional considerations," Working paper, University of Windsor, Canada, 1991.

123) Venugopal, V., "Artificial neural networks and fuzzy models: new tools for part-machine grouping," In Group Technology and Cellular Manufacturing (Editors: Suresh, N. C., and Kay, J. M.), Kluwer Academic Publishers, MA, 1998, pp. 169-184.

124) Chu, C. H., and Hayya, J. C., "A fuzzy clustering approach to manufacturing cell formation," International Journal of Production Research, Vol. 29, No. 8, 1991, pp. 1474-1487.

125) Gindy, N. N. Z., Ratchev, T. M., and Case, K., "Component grouping for GT applications – a fuzzy clustering approach with validity measure," International Journal of Production Research, Vol. 33, No. 9, 1995, pp. 2493-2509.

126) Wen, H. J., Smith, C. H., and Minor, E. D., "Formation and dynamic routing of part families among flexible manufacturing cells," International Journal of Production Research, Vol. 34, No. 8, 1996, pp. 2229-2245.

127) Leem, C. W., and Chen, J. J., "Fuzzy-set based machine-cell formation in cellular manufacturing," Journal of Intelligent Manufacturing, Vol. 7, No. 5, 1996, pp. 355-364.

128) Tsai, C. C., Manufacturing cell formation in a fuzzy environment, Ph. D. dissertation, Industrial and Manufacturing Systems Engineering Dept., Iowa State University, 1995.

129) Tsai, C. C., Chu, C. H., and Barta, A. T., "Modeling and analysis of manufacturing cell formation problem with fuzzy mixed-integer programming," IEE Transactions, Vol. 29, No. 7, 1997, pp. 533-547.

130) Zhang, C., and Wang, H., "Concurrent formation of part families and machine cells based on the fuzzy set theory," Journal of Manufacturing Systems, Vol. 11, No. 1, 1992, pp. 61-67.

131) Burke, L. I., and Kamal, S., "Fuzzy ART and cellular manufacturing," ANNIE 92 Conference Proceedings, 1992, pp. 779-784.

132) Burke, L. I., and Kamal, S., "Neural networks and the part family/machine group formation problem in cellular manufacturing: a framework using fuzzy ART," Journal of Manufacturing Systems, Vol. 14, No. 3, 1995, pp. 148-159.

133) Suresh, N. C., and Kaparthi, S., "Performance of Fuzzy ART neural network for group technology cell formation," International Journal of Production Research, Vol. 32, No. 7, 1994, pp. 1693-1713.

134) Kamal, S., and Burke, L. I., "FACT: a new neural network based clustering algorithm for group technology," International Journal of Production Research, Vol. 34, No. 4, 1996, pp. 919-946.

135) Dagli, C., Artificial neural networks for intelligent manufacturing, Chapman & Hall, London, 1993.

136) Zhang, H. C., and Huang, S. H., "Applications of neural networks in manufacturing: A state-of-the art survey," International Journal of Production Research, Vol. 33, No. 3, 1995, pp. 705-728.

137) Jamal, A. M. M., "Neural network and cellular manufacturing," Industrial Management & Data Systems, Vol. 93, No. 3, 1993, pp. 21-25.

138) Chakraborty, K., and Roy, U., "Connectionist models for part family classifications," Computers and Industrial Engineering, Vol. 24, 1993, pp. 189-198.

139) Kaparthi, S., and Suresh, N. C., "A neural network system for shape-based classification and coding of rotational parts," International Journal of Production Research, Vol. 29, 1991, pp. 1771-1784.

140) Kao, Y., and Moon, Y. B., "A unified group technology implementation using the back-propagation learning rule of neural network," Computers and Industrial Engineering, Vol. 20, 1991, pp. 425-437.

141) Arizono, I., Kato, M., Yamamoto, A., and Ohta, H., "A new stochastic neural network model and its application to grouping parts and tools in flexible manufacturing systems," International Journal of Production Research, Vol. 33, No. 6, 1996, pp. 1535-1548.

142) Chu, C. H., "An improved neural network for manufacturing cell formation," Decision Support Systems, Vol. 20, 1997, pp. 279-295.

143) Malave, C. O., and Ramachandran, S., "A neural network-based design of cellular manufacturing system," Journal of Intelligent Manufacturing, Vol. 2, 1991, pp. 305-314.

144) Venugopal, V., and Narendran, T. T., "A neural network approach for designing cellular manufacturing systems," Advances in Modeling and Analysis, Vol. 32, No. 2, 1992, pp. 13-26.

145) Chu, C. H., "Manufacturing cell formation by competitive learning," International Journal of Production Research, Vol. 31, 1993, pp. 829-843.

146) Venugopal, V., and Narendran, T. T., "Machine-cell formation through neural network models," International Journal of Production Research, Vol. 32, No. 9, 1994, pp. 2105-2116.

147) Malakooti, B., and Yang, Z., "A variable-parameter unsupervised learning clustering neural network approach with application to machine-part group formation," International Journal of Production Research, Vol. 33, No. 9, 1995, pp. 2395-2413.

148) Moon, Y. B., "An interactive activation and competition model for machine-part family formation in group technology," Proceedings of International Joint Conference on Neural Networks, Washington D. C., Vol. 2, 1990, pp. 667-670.

149) Moon, Y. B., and Chi, S. C., "Generalized part family formation using neural network techniques," Journal of Manufacturing Systems, Vol. 11, No. 3, 1992, pp. 149-159.

150) Currie, K. R., "An intelligent grouping algorithm for cellular manufacturing," Computers and Industrial Engineering, Vol. 23, 1992, pp. 109-112.

151) Kohonen, T., "Self-organization and associative memory," Springer-Verlag, Berlin, 1984.

152) Lee, H., Malave, C. O., and Ramachandran, S., "A self-organizing neural network approach for the design of cellular manufacturing systems," Journal of Intelligent Manufacturing, Vol. 3, 1992, pp. 325-332.

153) Rao, H. A., and Gu, P., "Expert self-organizing neural network for the design of cellular manufacturing systems," Journal of Manufacturing Systems, Vol. 13, No. 5, 1994, pp. 346-358.

154) Kiang, M. Y., Kulkarni, U. R., and Tam, K. Y., "Self-organizing map network as an interactive clustering tool – an application to group technology," Decision Support Systems, Vol. 15, No. 4, 1995, pp. 351-374.

155) Kulkarni, U. R., and Kiang, M. Y., "Dynamic grouping of parts in flexible manufacturing systems – a self-organizing neural network approach," European Journal of Operational Research, Vol. 84, No. 1, 1995, pp. 192-212.

156) Kusiak, A., and Chung, Y., "GT/ART: using neural networks to form machine cells," Manufacturing Review, Vol. 4, No. 4, 1991, pp. 293-301.

157) Dagli, C., and Sen, C. F., "ART1 neural network approach to large scale group technology problems," in Robotics and Manufacturing: Recent trends in research, education, and applications, ASME Press, NY, 1992, pp. 787-792.

158) Dagli, C., and Huggahalli, R., "Machine-part family formation with the adaptive resonance theory paradigm," International Journal of Production Research, Vol. 33, No. 4, 1995, pp. 893-913.

159) Chen, S. J., and Cheng, C. S., "A neural network-based cell formation algorithm in cellular manufacturing," International Journal of Production Research, Vol. 33, No.2, 1995, pp. 293-318.

160) Rao, H. A., and Gu, P., "A multi-constraint neural network for the pragmatic design of cellular manufacturing systems," International Journal of Production Research, Vol. 32, No. 7, 1995, pp. 1049-1070.

161) Enke, D., Ratanapan, K., and Dagli, C., "Machine-part family formation utilizing an ART1 neural network implemented on a parallel neuro-computer," Computers and Industrial Engineering, Vol. 34, No. 1, 1998, pp. 189-205.

162) Holland, J. H., Adaptation in neural and artificial systems, University of Michigan Press, MI, 1975.

163) Joines, J. A., King, R. E., and Culbreth, C. T., "Cell formation using genetic algorithm," In Group Technology and Cellular Manufacturing (Editors: Suresh, N. C., and Kay, J. M.), Kluwer Academic Publishers, MA, 1998, pp. 185-204.

164) Davis, L., and Coombs, S., "Optimizing network link sizes with genetic algorithms," In Modeling and Simulation Methodology: Knowledge Systems Paradigms, North Holland, Amsterdam, 1987.

165) Fourman, M. P., "Compaction of symbolic layout using genetic algorithms," Proceeding International Conference of Genetic Algorithms Applications, 1985, pp. 141-153.

166) Goldberg, D. E., "Computer-aided pipeline operation using genetic algorithms and rule learning," API Pipeline Cybernetics Symp., Houston, TX, 1984.

167) Schaffer, J. D., and Grefenstette, J. J., "Multi-objective learning via genetic algorithms," Proceeding 9th International Joint Conference of Artificial Intelligence, Los Angeles, 1985, pp. 593-595.

168) Venugopal, V., and Narendran, T. T., "A genetic algorithm approach to the machine-component grouping problem with multiple objectives," Computers and Industrial Engineering, Vol. 22, No. 4, 1992, pp. 469-480.

169) Gupta, Y. P., Gupta, M. C., Kumar, A., and Sundram, C., "Minimizing total intercell and intracell moves in cellular manufacturing: a genetic algorithm approach," International Journal of Computer Integrated Manufacturing, Vol. 9, 1995, pp. 92-101.

170) Gupta, Y. P., Gupta, M. C., Kumar, A., and Sundram, C., "A genetic algorithm-based approach to cell composition and layout design problems," International Journal of Production Research, Vol. 34, 1996, pp. 447-482.

171) Hsu, C. M., and Su, C. T., "Multi-objective machine-component grouping in cellular manufacturing: a genetic algorithm," Production Planning & Control, Vol. 9, No. 2, 1998, pp. 155-166.

172) Cheng, C. H., Lee, W. H., and Miltenburg, J., "A bi-chromosome genetic algorithm for minimizing intercell and intracell moves," In Group Technology and Cellular Manufacturing (Editors: Suresh, N. C., and Kay, J. M.), Kluwer Academic Publishers, MA, 1998, pp. 205-217.

173) Kamrani, A. K., *et al.*, "Simulation-based methodology for machine cell design," Computers and Industrial Engineering, Vol. 34, No. 1, 1998, pp. 173-188.

174) Chen, W. H., and Srivastava, B., "Simulated annealing procedures for forming machine cells in group technology," European Journal of Operational Research, Vol. 75, 1994, pp. 100-111.

175) Venugopal, V., and Narendran, T. T., "Cell formation in manufacturing systems through simulated annealing: an experimental evaluation," European Journal of Operational Research, Vol. 63, 1992, pp. 409-433.

176) Sofianopoulou, S., "Application of simulated annealing to a linear model for the formulation of machine cells in group technology," International Journal of Production Research, Vol. 35, No. 2, 1997, pp. 501-511.

177) Logendran, R., Ramakrishna, P., and Sriskandrajah, C., "Tabu search-based heuristics for cellular manufacturing systems in the presence of alternative process plans," International Journal of Production Research, Vol. 32, No. 2, 1994, pp. 273-297.

178) Logendran, R., and Ramakrishna, P., "Manufacturing cell formation in the presence of lot splitting and multiple units of the same machine," International Journal of Production Research, Vol. 33, No. 3, 1995, pp. 675-693.

179) Logendran, R., and Puvanunt, V., "Duplication of machines and subcontracting of parts in the presence of alternative cell location," Computers and Industrial Engineering, Vol. 33, Nos. 1-2, 1997, pp. 235-238.

180) Longendran, R., and Ko, C. S., "Manufacturing cell formation in the presence of flexible cell locations and material transporters," Computers and Industrial Engineering, Vol. 33, Nos. 3-4, 1997, pp. 545-548.

181) Rheault, M., Drolet, J. R., and Abdulnour, G., "Dynamic cellular manufacturing system (DCMS)," Computers and Industrial Engineering, Vol. 31, No.1, 1996, pp. 143-146.

182) Rheault, M., Drolet, J., and Abdulnour, G., "Physically reconfigurable virtual cells: a dynamic model for a highly dynamic environment," Computers and Industrial Engineering, Vol. 29, No. 1-4, 1995, pp. 221-225.

183) Kannan, V. R., "A simulation analysis of the impact of family configuration on virtual cellular manufacturing," Production Planning and Control, Vol. 8, No. 1, 1997, pp. 14-24.

184) Kannan, V. R., "Analyzing the trade-off between efficiency and flexibility in cellular manufacturing systems," Production Planning & Control, Vol. 9, No. 6, 1998, pp. 572-579.

185) Irani, S. A., Cavalier, T. M., and Cohen, P. H., "Virtual manufacturing cells: exploiting layout design and intercell flows for the machine-sharing problem," International Journal of Production Research, Vol. 31, No. 4, 1993, pp. 791-810.

186) Hsieh, L. F., and Sha, D. Y., "Heuristic algorithm for the design of facilities layout and AGV routes in tandem AGV systems," International Journal of Industrial Engineering, Vol. 4, No. 1, 1997, pp. 52-61.

187) Chachra, V., Ghare, P. M., and Moore, J. M., Applications of Graph Theory Algorithms, Elsevier North Holland Inc., NY, 1979.

188) Sugiyama, M., Sessomboon, W., Irohara, T., and Yoshimoto, K., "A design technique for AGV flow path by considering machine layout," The ASME International Mechanical Engineering Congress & Exposition, November 1997, pp. 1-8.

BIOGRAPHICAL SKETCH

Kuo-Cheng Ko, son of Ching-Chang Ko and Shin Kao, was born in April 25, 1967, in Shin-Tsu, Taiwan. He graduated from Chang-Hwa High School in 1985. Mr. Ko received the Bachelor of Science in Industrial Engineering from Feng-Chia University, Taiwan, in 1990. For the following two years, he served for Taiwan Army as a sergeant. Mr. Ko joined the Department of Industrial Engineering at Da-Yeh University, Taiwan, as a teaching faculty in 1992.

In 1994, Mr. Ko attended Iowa State University. He earned the Master of Science in Industrial Engineering in 1996 and continued for doctoral study. Currently, Mr. Ko is a PH.D. candidate in Industrial Engineering in the Department of Industrial & Manufacturing Systems Engineering at Iowa State University.

Mr. Ko has served as a Teaching Assistant in the Department of Industrial & Manufacturing Systems Engineering at Iowa State University. His areas of interest are material handling, production planning and control, analysis of manufacturing system, and information engineering. His professional affiliations include the Institute of Industrial Engineers and INFORMS.